RESOURCE AND PATIENT MANAGEMENT SYSTEM

Patient Chart GUI Program

TECHNICAL MANUAL

Version 1.2
December 2001

Information Technology Support Center
Division of Information Resources
Albuquerque, New Mexico

PREFACE

The purpose of this manual is to provide technical information about the Patient Chart GUI package (BPC). The IHS Patient Chart program is a Windows-based Graphical User Inteface (GUI) application that allows a provider to review and, in some cases, edit or add patient data. Currently the application contains problems, medications, allergies, labs, purpose of visit, measurements, Diabetic Patient Care Summary, referrals, appointments, x-ray, patient education, Women's Health, Health Summeries, immunizations, and telnet capability.

POINT OF CONTACT

For Software Support, please contact your Area ISC or the ITSC Support Center listed below:

Division of Information Resources Software Support Center
Telephone: (505) 248-4371 or (888) 830-7280
Fax: (505) 248-4199

TABLE OF CONTENTS

1.0 INTRODUCTION

The IHS GUI Patient Chart Program is a fully functional system developed by the Indian Health Service (IHS) and interfaces with the RPMS (Resource and Patient Management System) package suite. The IHS Patient Chart program is a Windows-based Graphical User Inteface (GUI) application that allows a provider to review and, in some cases, edit or add patient data. Currently the application contains Problems, Medications, Allergies, Labs, Purpose of Visit, Measurements, Diabetic Patient Care Summary, referrals, Appointments, X-ray, Patient Education, Women's Health, Health Summeries, Immunizations, and Telnet capability. It operates on computers at IHS or tribal administrative offices.

2.0 Server and Client Resource Requirements

2.1 Client-Side Resource And Software Requirements

A preliminary estimate of resource requirements is as follows:

Client Hardware (Minimum)

- IBM compatible computer
- Pentium III Intel-based processor 200mHz or higher with 32mb of RAM
- Windows 9X, NT, or 2000 operating system
- 15" VGA or SVGA monitor
- 800-megabyte Hard Drive
- Ethernet Card (16-bit or better)
- Printer compatible with hardware configuration (optional)
- Mouse
- 101-key Keyboard

Client Hardware (Preferred)

- IBM compatible computer
- Pentium III or greater with 128 mb or more of RAM
- NT or Window 2000 operating system.
- 17" VGA or SVGA monitor
- 4.0 Gigabyte or larger Hard Drive
- 10/100 Ethernet Card (PCI or 16-bit)
- Printer compatible with hardware configuration (optional)
- Mouse
- 101-key Keyboard

A wide variety of IBM compatible computers have been tested and the speed of the processor is the primary concern. The client software has also been tested on a number of laptops.

Client Software (Minimum)

- Microsoft Windows NT or 2000 operating system (NT with SP6 or 2000 with SP2)
- TCP Aware software (included in Windows 9x, NT, and 2000)
- Windows Internet Explorer 5.0 or above **(Setup may fail if you have an earlier version)**
- Internet IP address and domain name

2.2 Server-Side Resource And Software Requirements

RISC 6000 Standard IHS Configuration

- Properly installed TCPIP
- FIFO 72 or higher for AIX 4.33 or higher
- FIFO 69 or higher for AIX 4.2
- MSM 4.4.0 or higher
- Kernel 8.0 or higher
- FileMan 21 or higher
- BGU RPMS software 1.2 or higher

Note: A MUSERVER process must be defined for each TCP connection on the RISC platform and requires shutting down MSM to change the setting. It is projected that the Patient Chart application will require up to 20% additional RPMS Server resources when the application is fully functional.

Windows NT Server 4.0 with service patch 6A or higher

- MSM For Windows NT 4.4.1 or higher
- FIFO 10 for NT
- Kernel 8.0 or higher
- FileMan 21 or higher
- BGU RPMS software 1.2 or higher

Windows 2000 with service patch 2 or higher

- MSM For Windows NT 4.41 or higher
- FIFO 10 for NT
- Kernel 8.0 or higher
- FileMan 21 or higher
- BGU RPMS software 1.2 or higher

3.0 PACKAGE-WIDE VARIABLES

There are no package-wide variables associated with the Patient Chart GUI Program.

3.1 Implementation And Maintenance

There are no action items that need to be accomplished by the Installer or Program Manager after the init process.

4.0 ROUTINES AND SECURITY

4.1 Routines

The following is a listing of the routines exported with the BPC package.

FIRST LINE LIST:BGU*

```
BGU1200E  ; IHS/ITSC/FJE,MJL - BGU ENVIRONMENT CHECK ROUTINE ;
          ;;1.2;BGU;;SEP 25, 2001
BGUAPI    ; IHS/ITSC/FJE,MJL - Call back for API/RPC Query Utility ;
          ;;1.2;BGU;;SEP 25, 2001
BGUCEKT   ; IHS/ITSC/FJE,MJL - GENERAL PATIENT LOOKUP FOR BGU WINDOWS ;
          ;;1.2;BGU;;SEP 25, 2001
BGUCND    ; IHS/ITSC/FJE,MJL - CONDITION HANDLER ;
          ;;1.2;BGU;;SEP 25, 2001
BGUCU     ; IHS/ITSC/FJE,MJL - CLEAN UP BGU TRACE ENTRIES ;
          ;;1.2;BGU;;SEP 25, 2001
BGUDSTR   ; IHS/ITSC/FJE,MJL - DESTRUCTOR FOR MDAO ;  [ 09/27/2001   1:05 PM ]
          ;;1.2;BGU;;SEP 25, 2001
BGUFILE   ; IHS/ITSC/FJE,MJL - CALLS FILE RECORDS ;
          ;;1.2;BGU;;SEP 25, 2001
BGUFLR    ; IHS/ITSC/FJE,MJL - BGU FILER ;
          ;;1.2;BGU;;SEP 25, 2001
BGUGDE    ; IHS/ITSC/FJE,MJL - GET FILE DICTIONARY ELEMENTS ;
          ;;1.2;BGU;;SEP 25, 2001
BGUGFAC   ; IHS/ITSC/FJE,MJL - GETS FACILITIES FOR A USER ;
          ;;1.2;BGU;;SEP 25, 2001

BGUGPLK   ; IHS/ITSC/FJE,MJL - GENERAL PATIENT LOOKUP FOR BGU WINDOWS ;
          ;;1.2;BGU;;SEP 25, 2001
BGUGUTL   ; IHS/ITSC/FJE,MJL - GENERAL UTILITIES FOR GUI LOGOFF ;
          ;;1.2;BGU;;SEP 25, 2001
BGULIST   ; IHS/ITSC/FJE,MJL - GENERAL FILE LISTER ;  [ 09/27/2001   1:05 PM ]
          ;;1.2;BGU;;SEP 25, 2001
BGULIST1  ; IHS/ITSC/FJE,MJL - GENERAL FILE LISTER ;
          ;;1.2;BGU;;SEP 25, 2001
BGULIST2  ; IHS/ITSC/FJE,MJL - GENERAL FILE LISTER ;  [ 09/27/2001   1:05 PM ]
          ;;1.2;BGU;;SEP 25, 2001
BGUPMR    ; IHS/ITSC/FJE,MJL - PRINT MUMPS ROUTINES ;  [ 09/27/2001  1:28 PM ]
          ;;1.2;BGU;;SEP 25, 2001
BGUPOST   ; IHS/ITSC/FJE,MJL - BGU TRACE FILE CLEANUP ;
          ;;1.2;BGU;;SEP 25, 2001
BGUPSRPC  ; IHS/ITSC/FJE,MJL - GENERAL BGU UTILITY ;
          ;;1.2;BGU;;SEP 25, 2001
BGUPURGE  ; IHS/ITSC/FJE,MJL - PURGE ROUTINE FOR BGUSEC ;
```

```
        ;;1.2;BGU;;SEP 25, 2001
BGURTNFL  ; IHS/ITSC/FJE,MJL - SILENT ROUTINE FIRST LINE ;
        ;;1.2;BGU;;SEP 25, 2001
BGUTCP    ; IHS/ITSC/FJE,MJL - Primary Control TCP listener ;  [ 10/01/2001  9:00 AM ]
        ;;1.2;BGU;;SEP 25, 2001
BGUTCPH   ; IHS/ITSC/FJE,MJL - Service TCP Messages ;  [ 09/27/2001  2:18 PM ]
        ;;1.2;BGU;;SEP 25, 2001
BGUTCPL   ; IHS/ITSC/FJE,MJL - Background Listener for TCP connects ;  [ 10/01/2
001  9:03 AM ]
        ;;1.2;BGU;;SEP 25, 2001
BGUTRACE  ; IHS/ITSC/FJE,MJL - DEBUGGING TRACE FACILITY ;
        ;;1.2;BGU;;SEP 25, 2001
BGUTRACS  ; IHS/ITSC/FJE,MJL - REMOTE PROCEDURE TESTING ROUTINE ;  [
09/27/2001
 1:50 PM ]
        ;;1.2;BGU;;SEP 25, 2001
BGUTRCU   ; IHS/ITSC/FJE,MJL - Clean up routine for the Debug/Trace global:
^BGUTRACE ;
        ;;1.2;BGU;;SEP 25, 2001
BGUTRLU   ; IHS/ITSC/FJE,MJL - DEBUGGING TRACE FACILITY ;
        ;;1.2;BGU;;SEP 25, 2001
BGUTUTL   ; IHS/ITSC/FJE,MJL - Trace utilities ;
        ;;1.2;BGU;;SEP 25, 2001
BGUXUSRB  ; IHS/ITSC/FJE,MJL - Request Broker ;  [ 10/01/2001  8:51 AM ]
        ;;1.2;BGU;;SEP 25, 2001
BGUXUSRC  ; IHS/ITSC/FJE,MJL - Request Broker ;  [ 10/01/2001  8:46 AM ]
        ;;1.2;BGU;;SEP 25, 2001

          30 ROUTINES

--------------------------------------------------------------------------
```

FIRST LINE LIST: BPC*

```
BPC1200E  ; IHS/ITSC/FJE,MJL - BPC PATIENT CHART INITIAL INSTALL
ENVIRONMENT CHE
CK ;  [ 09/25/2001  9:58 AM ]
        ;;1.2;BPC;;SEP 25, 2001
BPC7OGC   ; IHS/ITSC/FJE,MJL - Interim report rpc chart 8/1/97 12:12 ;  [ 09/25/2001  12:35
PM ]
        ;;1.2;BPC;;SEP 25, 2001
BPC7OGG   ; IHS/ITSC/FJE,MJL - Interim report rpc grid 6/23/97 15:34 ;  [ 09/25/2001  12:35
PM ]
        ;;1.2;BPC;;SEP 25, 2001
BPC7OGM   ; IHS/ITSC/FJE,MJL - Interim report rpc memo 10/15/98 22:21 ;  [ 09/25/2001
12:35 PM ]
        ;;1.2;BPC;;SEP 25, 2001
```

```
BPC7OGMC  ; IHS/ITSC/FJE,MJL - Interim report rpc memo chem 5/20/97 19:03 ;  [
09/25/2001  12:35 PM ]
        ;;1.2;BPC;;SEP 25, 2001
BPC7OGMG  ; IHS/ITSC/FJE,MJL - Interim report rpc memo grid 5/22/97 14:23 ;  [
09/25/2001  12:35 PM ]
        ;;1.2;BPC;;SEP 25, 2001
BPC7OGMM  ; IHS/ITSC/FJE,MJL - Interim report rpc memo micro 5/20/97 18:52 ;
[09/25/2001  12:35 PM ]
        ;;1.2;BPC;;SEP 25, 2001
BPC7OGMP  ; IHS/ITSC/FJE,MJL - Interim report rpc memo print 5/22/97 18:40 ;  [09/25/2001
12:35 PM ]
        ;;1.2;BPC;;SEP 25, 2001
BPC7OGMU  ; IHS/ITSC/FJE,MJL - Interim report rpc memo utility 6/6/97 16:15 ;
        ;;1.2;BPC;;SEP 25, 2001
BPC7OGO   ; IHS/ITSC/FJE,MJL - Interim report rpc other 12/12/97 14:22 ;
        ;;1.2;BPC;;SEP 25, 2001
BPC7OGU   ; IHS/ITSC/FJE,MJL - Interim report rpc utility 5/22/97 13:53 ;  [ 09/
25/2001  12:35 PM ]
        ;;1.2;BPC;;SEP 25, 2001
BPCALRG   ; IHS/ITSC/FJE,MJL - ALLERGY TRACKING RPC ROUTINE - FHL ;
        ;;1.2;BPC;;SEP 25, 2001
BPCAPPT   ; IHS/ITSC/FJE,MJL - GUI APPT LIST ;
        ;;1.2;BPC;;SEP 25, 2001
BPCBWDSP  ; IHS/ITSC/FJE,MJL - DISPLAY PATIENT PROFILE ;
        ;;1.2;BPC;;SEP 25, 2001
BPCDBS    ; IHS/ITSC/FJE,MJL - DIABETIC SUMMARY UTILITY FOR GUI PAT CHART ;
        ;;1.2;BPC;;SEP 25, 2001
BPCDBS1   ; IHS/ITSC/FJE,MJL - DIABETIC CARE SUMMARY SUPPLEMENT ;
        ;;1.2;BPC;;SEP 25, 2001
BPCGDX    ; IHS/ITSC/FJE,MJL - PROGRAM TO GET LIST OF DIAGNOSES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCHSDSP  ; IHS/ITSC/FJE,MJL - HEALTH SUMMARY DISP GUI RPC ROUTINE ;
        ;;1.2;BPC;;SEP 25, 2001
BPCIMDSP  ; IHS/ITSC/FJE,MJL - IMMUN HISTORY DISP GUI RPC ROUTINE ;
        ;;1.2;BPC;;SEP 25, 2001
BPCLAB    ; IHS/ITSC/FJE,MJL - GUI LAB INTERIM ;  [ 09/25/2001  12:39 PM ]
        ;;1.2;BPC;;SEP 25, 2001
BPCLALL   ; IHS/ITSC/FJE,MJL - PCC VISIT LIST FOR GUI ;
        ;;1.2;BPC;;SEP 25, 2001
BPCLALL1  ; IHS/ITSC/FJE,MJL - CONT BPCLALL LOAD INIT DATA PAT CHART ;
        ;;1.2;BPC;;SEP 25, 2001
BPCLIPL   ; IHS/ITSC/FJE,MJL - GUI PROVIDER PAT LIST FOR LAB DATA ;
        ;;1.2;BPC;;SEP 25, 2001
BPCLOPRT  ; IHS/ITSC/FJE,MJL - PRINT LAB ORDERS FOR GUI ;
        ;;1.2;BPC;;SEP 25, 2001
BPCLORD   ; IHS/ITSC/FJE,MJL - GUI CHART LAB ORDER SCREEN ;
```

```
        ;;1.2;BPC;;SEP 25, 2001
BPCLRML   ; IHS/ITSC/FJE,MJL - GUI GET MY LABS FOR DATE RANGE ;  [ 09/25/2001
12:49 PM ]
        ;;1.2;BPC;;SEP 25, 2001
BPCMEAS   ; IHS/ITSC/FJE,MJL - GUI V MEASUREMENT VISIT CREATION ;  [
09/25/2001 11:01 AM ]
        ;;1.2;BPC;;SEP 25, 2001
BPCNOTEC  ; IHS/ITSC/FJE,MJL - FHL-6/14/96 - ADD,DELETE OR EDIT PROBLEM
NOTE ;
        ;;1.2;BPC;;SEP 25, 2001
BPCPC     ; IHS/ITSC/FJE,MJL - PATIENT CHART RELATED ROUTINES FOR GUI ;
        ;;1.2;BPC;;SEP 25, 2001
BPCPC1    ; IHS/ITSC/FJE,MJL - PATIENT CHART GUI ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCPC2    ; IHS/ITSC/FJE,MJL - PATIENT CHART GUI ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCPRBC   ; IHS/ITSC/FJE,MJL - ADD,DELETE, OR EDIT PROBLEM ;
        ;;1.2;BPC;;SEP 25, 2001
BPCPROB   ; IHS/ITSC/FJE,MJL - GUI COLLECTS PROBLEM LIST DATA ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC     ; IHS/ITSC/FJE,MJL - REFERRED CARE GUI RPC ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC1    ; IHS/ITSC/FJE,MJL - REFERRED CARE GUI RPC ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC2    ; IHS/ITSC/FJE,MJL - REFERRED CARE GUI RPC ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC3    ; IHS/ITSC/FJE,MJL - 12/03/96 -REFERRED CARE ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC4    ; IHS/ITSC/FJE,MJL - FHL-12/5/96 - REFERRED CARE GUI ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC5    ; IHS/ITSC/FJE,MJL - FHL-12/11/96 - REFERRED CARE GUI ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC6    ; IHS/ITSC/FJE,MJL - FHL-12/26/96 - REFERRED CARE GUI ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC7    ; IHS/ITSC/FJE,MJL - REFERRED CARE GUI RPC ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC8    ; IHS/ITSC/FJE,MJL - REFERRED CARE GUI RPC ROUTINES ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRC9    ; IHS/ITSC/FJE,MJL - REFERRED CARE GUI RPC ROUTINES ;  [ 09/25/2001
11:01 AM ]
        ;;1.2;BPC;;SEP 25, 2001
BPCRXNXT  ; IHS/ITSC/FJE,MJL - FINDS NEXT PERSCRIPTION NUMBER FOR BPC
GUI ;
        ;;1.2;BPC;;SEP 25, 2001
BPCRXSIG  ; IHS/ITSC/FJE,MJL - EXPANDS SIG FOR BPC GUI ;
        ;;1.2;BPC;;SEP 25, 2001
```

```
BPCSC     ; IHS/ITSC/FJE,MJL - SPECIAL CROSS REFERENCE ROUTINE ;
          ;;1.2;BPC;;SEP 25, 2001
BPCSC1    ; IHS/ITSC/FJE,MJL - SPECIAL X-REF ROUTINES - ;  [ 09/25/2001  11:14 A
M ]
          ;;1.2;BPC;;SEP 25, 2001


            47 ROUTINES
```

4.2 Security Keys

The Security Keys are assigned based on the role of the user, just as in the traditional RPMS applications. Depending on the access needs of your site's users, your security key assignments may be very similar or very dissimilar between staff levels. The chart below is a guide for assigning security keys for the Patient Chart program. The chart below has been left blank so that you can keep a record of your site's access levels and the associated security keys for future reference. If you need assistance in determining what keys to assign to your site staff, please contact the ITSC Support Center at (888) 830-7280 or rpmshelp@mail.ihs.gov.

Tip: Because of the number of keys that need to be assigned, we recommend that you allocate all keys to all users and then manually deallocate the keys that are inappropriate for each individual user.

4.2.1 Security Key Allocation Table

BPC package keys which control GUI functionality are listed below. A descriptive list follows.

Security Key	Descriptive Name	Provider	Nurse	Clerk	Lab	Pharmacy	Social
TABS							
BPCTAP	APPTS						
BPCTCV	COVER						
BPCTEP	EDUCATION PROTOCOL						
BPCTFS	FACE SHEET						
BPCTHS	HEALTH SUM						
BPCTIM	IMMUNIZATIONS						
BPCTLR	LAB						
BPCTMS	MEASUREMENT						
BPCTPC	PCC OTHER						
BPCTPR	PROBLEM						
BPCTPV	POV						
BPCTRG	REG						
BPCTRX	MEDICATIONS						
BPCTSP	SPECIALS						
BPCTTN	TELNET						
BPCTWH	WOMEN HEALTH						
BPCTXR	XRAY						
CHART							
BPCPC	MASTER						
PATIENT							
BPCPATPL	PAT PICKLIST						
MAIN							
BPCMNML	MY LABS						
BPCMNMLP	MY LABS PRINT						
BPCMNMLR	MY LABS REFRSH						
BPCMNPLU	PAT LOOKUP						

Security Key	Descriptive Name	Provider	Nurse	Cler k	Lab	Pharmacy	Social
BPCMNPO	PAT OPTIONS						
BPCMNTN	TELNET						
BPCMNUO	USER OPTIONS						
FACE SHEET							
BPCFSP	PRINT						
PROBLEM							
BPCPRA	ADD						
BPCPRD	DEL						
BPCPRE	EDIT						
BPCPRL	LIST						
BPCPRNA	NOTE ADD						
BPCPRNC	NOTE CHANGE						
BPCPRNR	NOTE REMOVE						
BPCPRP	PRINT						
PHARMACY							
BPCRXMP	MED PROFILE						
BPCRXMPD	MED PR DATE						
BPCRXMPP	MED PR PRT						
BPCRXMPR	MED PR REFRESH						
BPCRXPMD	PCC MED DATE						
BPCRXPMP	PCC MED PRINT						
BPCRXPMR	PCC MED REFRSH						
LAB							
BPCLIL	INTERIM						
BPCLILC	INT DATE CHG						
BPCLILP	INT PRINT						
BPCLILR	INT REFRESH						
BPCLM	MICRO						
BPCLO	ORDER						
BPCLOCM	ORDER COMMS						
BPCLOLA	ACCESSION						
BPCLOLC	COLLECTION						
BPCLOLO	ORD LOCATION						
BPCLOOO	ORD OTHER ORD						
BPCLOPO	ORD PLACE ORD						
BPCLOSP	SEND PAT						
BPCLOWC	WARD COLL						
BPCLP	PRINT						
BPCLT	TRENDS						
BPCLTE	TREND EXCEL						
BPCLTP	TREND PRINT						
MEASUREMENTS							
BPCMSA	ADD						
BPCMSE	EDIT						
BPCMSG	GRAPH						
BPCMSS	SAVE						
BPCMSTP	GRAPH PRINT						
SPECIALS							
BPCSPDA	ALLERGY						

Security Key	Descriptive Name	Provider	Nurse	Clerk	Lab	Pharmacy	Social
BPCSPDS	DIABETES						
BPCSPRC	REF CARE						
BPCSPTR	TROUBLESHOOT						
BPCSPTRP	TRBSHT PRINT						
TELNET							
BPCNTNH	TN NEW HOST						
HEALTH SUMM							
BPCHSD	DISPLAY						
BPCHSP	PRINT						
BPCHST	TYPE						
RADIOLOGY							
BPCXRP	PRINT						
WOMEN'S HLTH							
BPCBWD	TAB						
BPCBWP	PRINT						
APPOINTMENT							
BPCSDP	PRINT						
IMMUNIZATIONS							
BPCIMDF	DSP FORECAST						
BPCIMDH	DSP HISTORY						
PAT EDUCATION							
BPCEDD	DISPLAY						
BPCEDP	PRINT						

4.2.2 Security Key Descriptions

NAME: BGUZMENU
DESCRIPTIVE NAME: BGU Main Menu Key
DESCRIPTION: This key is assigned to the users who need to monitor the BGU Remote Procedure file and security.

NAME: BPCBWD
Descriptive Name: PAT CHART WOMEN HLTH DISPLAY
DESCRIPTION: Key necessary to access WO HLTH DSP option in the GUI PAT CHART program.

NAME: BPCBWP
DESCRIPTIVE NAME: PAT CHART WOMEN HLTH PRINT
DESCRIPTION: Key necessary to access WO HLTH PRINT option in the GUI PAT CHART program.

NAME: BPCEDD
DESCRIPTIVE NAME: PAT CHART ED PROTOCOL DISPLAY
DESCRIPTION: Key necessary to access ED PROTOCOL DISP option in the GUI PAT CHART program.

NAME: BPCEDP
DESCRIPTIVE NAME: PAT CHART ED PROTOCOL PRINT
DESCRIPTION: Key necessary to access ED PROTOCOL PRINT option in the GUI PAT CHART program.

NAME: BPCFSP DESCRIPTIVE NAME: PAT CHART FACE PRINT
DESCRIPTION: Key necessary to access FACE SHEET PRINT option in the GUI PATIENT CHART program

NAME: BPCHSD
DESCRIPTIVE NAME: PAT CHART HLTH SUM DISPLAY
DESCRIPTION: Key necessary to access HLTH SUM DSP option in the GUI PATIENT CHART program

NAME: BPCHSP
DESCRIPTIVE NAME: PAT CHART HLTH SUM PRINT
DESCRIPTION: Key necessary to access HLTH SUM PRT option in the GUI PATIENT CHART program

NAME: BPCHST
DESCRIPTIVE NAME: PAT CHART HLTH SUM TYPE
DESCRIPTION: Key necessary to access HLTH SUM TYP option in the GUI PATIENT CHART program

NAME: BPCIMA
DESCRIPTIVE NAME: PAT CHART IMM ADD
DESCRIPTION: Key necessary to access IMM ADD option in the GUI PATIENT CHART program

NAME: BPCIMD
DESCRIPTIVE NAME: PAT CHART IMM DELETE
DESCRIPTION: Key necessary to access IMM DELETE option in the GUI PATIENT CHART program

NAME: BPCIMDF
DESCRIPTIVE NAME: PAT CHART IMM DSP FORECAST
DESCRIPTION: Key necessary to access IMM DSP FORCAST option in the GUI PATIENT CHART program

NAME: BPCIMDH
DESCRIPTIVE NAME: PAT CHART IMM DSP HISTORY
DESCRIPTION: Key necessary to access IMM DSP HISTORY option in the GUI PATIENT CHART program

NAME: BPCIME
DESCRIPTIVE NAME: PAT CHART IMM EDIT

DESCRIPTION: Key necessary to access IMM EDIT option in the GUI PATIENT CHART program

NAME: BPCLIL
DESCRIPTIVE NAME: PAT CHART LAB INTERIM DESCRIPTION: Key necessary to access INTERIM LABS option in the GUI PATIENT CHART program

NAME: BPCLILC
DESCRIPTIVE NAME: PAT CHART LAB INT DATE CHG
DESCRIPTION: Key necessary to access INT LABS DATE CHG option in the GUI PATIENT CHART program

NAME: BPCLILP
DESCRIPTIVE NAME: PAT CHART LAB INT PRINT
DESCRIPTION: Key necessary to access INT LABS PRINT option in the GUI PATIENT CHART program

NAME: BPCLILR
DESCRIPTIVE NAME: PAT CHART LAB INT REFRESH
DESCRIPTION: Key necessary to access INT LABS REFRESH option in the GUI PATIENT CHART program

NAME: BPCLM
DESCRIPTIVE NAME: PAT CHART LAB MICRO
DESCRIPTION: Key necessary to access LAB MICRO option in the GUI PATIENT CHART program

NAME: BPCLO
DESCRIPTIVE NAME: PAT CHART LAB ORDER
DESCRIPTION: Key necessary to access LAB ORDER option in the GUI PATIENT CHART program

NAME: BPCLOCM
DESCRIPTIVE NAME: PAT CHART LAB ORDER COMMENTS
DESCRIPTION: Key necessary to access LAB ORDER COMMENTS option in the GUI PATIENT CHART program

NAME: BPCLOLA
DESCRIPTIVE NAME: PAT CHART LAB ACCESSION
DESCRIPTION: Key necessary to access LAB ORDER LAB ACCESSIONING option in the GUI PATIENT CHART program

NAME: BPCLOLC
DESCRIPTIVE NAME: PAT CHART LAB COLLECTION
DESCRIPTION: Key necessary to access LAB ORDER LAB COLLECT option in the GUI PATIENT CHART program

NAME: BPCLOLO
DESCRIPTIVE NAME: PAT CHART LAB ORD LOCATION
DESCRIPTION: Key necessary to access LAB ORDER LOCATION option in the GUI PATIENT CHART program

NAME: BPCLOOO
DESCRIPTIVE NAME: PAT CHART LAB ORD OTHER ORDER
DESCRIPTION: Key necessary to access LAB ORDER OTHER ORDER option in the GUI PATIENT CHART program

NAME: BPCLOPO
DESCRIPTIVE NAME: PAT CHART LAB ORD PLACE ORDER
DESCRIPTION: Key necessary to access LAB ORDER PLACE ORDER option in the GUI PATIENT CHART program

NAME: BPCLOSP
DESCRIPTIVE NAME: PAT CHART LAB SEND PAT
DESCRIPTION: Key necessary to access LAB ORDER SEND PATIENT option in the GUI PATIENT CHART program

NAME: BPCLOWC
DESCRIPTIVE NAME: PAT CHART LAB WARD COLL
DESCRIPTION: Key necessary to access LAB ORDER WARD/CLINIC COLLECT option in the GUI PATIENT CHART program

NAME: BPCLP
DESCRIPTIVE NAME: PAT CHART LAB PRINT
DESCRIPTION: Key necessary to access LAB PRINT option in the GUI PATIENT CHART program

NAME: BPCLT
DESCRIPTIVE NAME: PAT CHART LAB TRENDS
DESCRIPTION: Key necessary to access LAB TRENDS option in the GUI PATIENT CHART program

NAME: BPCLTE
DESCRIPTIVE NAME: PAT CHART LAB TREND EXCEL
DESCRIPTION: Key necessary to access LAB TREND EXCEL option in the GUI PATIENT CHART program

NAME: BPCLTP
DESCRIPTIVE NAME: PAT CHART LAB TREND PRINT
DESCRIPTION: Key necessary to access LAB TREND PRINT option in the GUI PATIENT CHART prog.

NAME: BPCMNML
DESCRIPTIVE NAME: PAT CHART MAIN MY LABS
DESCRIPTION: Key necessary to access MY LABS option in the GUI PATIENT CHART program

NAME: BPCMNMLP
DESCRIPTIVE NAME: PAT CHART MAIN MY LABS PRINT
DESCRIPTION: Key necessary to access MY LABS PRINT option in the GUI PATIENT CHART program

NAME: BPCMNMLR
DESCRIPTIVE NAME: PAT CHART MAIN MY LABS REFRESH
DESCRIPTION: Key necessary to access MY LABS REFRESH option in the GUI PATIENT CHART program

NAME: BPCMNPLU
DESCRIPTIVE NAME: PAT CHART MAIN PAT LOOKUP
DESCRIPTION: Key necessary to access PAT LOOKUP option in the GUI PATIENT CHART program

NAME: BPCMNPO
DESCRIPTIVE NAME: PAT CHART MAIN PAT OPTIONS
DESCRIPTION: Key necessary to access MAIN PAT OPTIONS option in the GUI PATIENT CHART program

NAME: BPCMNTN
DESCRIPTIVE NAME: PAT CHART MAIN TELNET
DESCRIPTION: Key necessary to access TELNET option in the GUI PATIENT CHART program

NAME: BPCMNUO
DESCRIPTIVE NAME: PAT CHART MAIN USER OPTIONS
DESCRIPTION: Key necessary to access USER OPTIONS option in the GUI PATIENT CHART program

NAME: BPCMSA
DESCRIPTIVE NAME: PAT CHART MEAS ADD
DESCRIPTION: Key necessary to access MEASUREMENT ADD option in the GUI PATIENT CHART program

NAME: BPCMSE
DESCRIPTIVE NAME: PAT CHART MEAS EDIT
DESCRIPTION: Key necessary to access MEASUREMENT EDIT option in the GUI PATIENT CHART program

NAME: BPCMSG

DESCRIPTIVE NAME: PAT CHART MEAS GRAPH
DESCRIPTION: Key necessary to access MEASUREMENT GRAPH option in the GUI PATIENT CHART program

NAME: BPCMSS
DESCRIPTIVE NAME: PAT CHART MEAS SAVE
DESCRIPTION: Key necessary to access MEASUREMENT SAVE option in the GUI PATIENT CHART program

NAME: BPCMSTP
DESCRIPTIVE NAME: PAT CHART MEAS GRAPH PRINT
DESCRIPTION: Key necessary to access MEASUREMENT GRAPH PRT option in the GUI PATIENT CHART program

NAME: BPCNTNH
DESCRIPTIVE NAME: PAT CHART TN NEW HOST
DESCRIPTION: Key necessary to access TELNET NEW HOST option in the GUI PATIENT CHART prog.

NAME: BPCPATPL
DESCRIPTIVE NAME: PAT CHART PAT PICK LIST
DESCRIPTION: Key necessary to access PAT PICK LIST option in the GUI PATIENT CHART program

NAME: BPCPC
DESCRIPTIVE NAME: PAT CHART MASTER
DESCRIPTION: Key necessary to access the GUI PATIENT CHART program

NAME: BPCPRA
DESCRIPTIVE NAME: PAT CHART PROB ADD
DESCRIPTION: Key necessary to access PROB ADD option in the GUI PATIENT CHART program

NAME: BPCPRD
DESCRIPTIVE NAME: PAT CHART PROB DEL
DESCRIPTION: Key necessary to access PROB DELETE option in the GUI PATIENT CHART program

NAME: BPCPRE
DESCRIPTIVE NAME: PAT CHART PROB EDIT
DESCRIPTION: Key necessary to access PROBLEM EDIT option in the GUI PATIENT CHART program

NAME: BPCPRL
DESCRIPTIVE NAME: PAT CHART PROB LIST

DESCRIPTION: Key necessary to access PROBLEM PRINT LIST option in the GUI PATIENT CHART program

NAME: BPCPRNA
DESCRIPTIVE NAME: PAT CHART PR NOTE ADD
DESCRIPTION: Key necessary to access PROB NOTE ADD option in the GUI PATIENT CHART program

NAME: BPCPRNC
DESCRIPTIVE NAME: PAT CHART PR NOTE CHG
DESCRIPTION: Key necessary to access PROBLEM NOTE CHANGE option in the GUI PATIENT CHART program

NAME: BPCPRNR
DESCRIPTIVE NAME: PAT CHART PR NOTE REM
DESCRIPTION: Key necessary to access PROBLEM NOTE REMOVE option in the GUI PATIENT CHART program

NAME: BPCPRP
DESCRIPTIVE NAME: PAT CHART PROB PRINT
DESCRIPTION: Key necessary to access PROBLEM PRINT option in the GUI PATIENT CHART program

NAME: BPCPVP
DESCRIPTIVE NAME: PAT CHART POV PRINT
DESCRIPTION: Key necessary to access POV PRINT option in the GUI PATIENT CHART program

NAME: BPCRXC
DESCRIPTIVE NAME: PAT CHART RX CANCEL
DESCRIPTION: Key necessary to access MEDS RX CANCEL option in the GUI PATIENT CHART program

NAME: BPCRXE
DESCRIPTIVE NAME: PAT CHART RX EDIT
DESCRIPTION: Key necessary to access MEDS RX EDIT option in the GUI PATIENT CHART program

NAME: BPCRXI
DESCRIPTIVE NAME: PAT CHART RX REINIAT
DESCRIPTION: Key necessary to access MEDS RX re-iNIAT option in the GUI PATIENT CHART program

NAME: BPCRXMP
DESCRIPTIVE NAME: PAT CHART RX MED PROFILE

DESCRIPTION: Key necessary to access MEDS RX MED PROFILE option in the GUI PATIENT CHART program

NAME: BPCRXMPD
DESCRIPTIVE NAME: PAT CHART RX MED PR DATE
DESCRIPTION: Key necessary to access MEDS RX MED PROFILE DATE option in the GUI PATIENT CHART program

NAME: BPCRXMPP
DESCRIPTIVE NAME: PAT CHART RX MED PR PRT
DESCRIPTION: Key necessary to access MEDS RX MED PROFILE PRINT option in the GUI PATIENT CHART program

NAME: BPCRXMPR
DESCRIPTIVE NAME: PAT CHART RX MED PR REFRESH
DESCRIPTION: Key necessary to access MEDS RX MED PROFILE REFRESH option in the GUI PATIENT CHART program

NAME: BPCRXN
DESCRIPTIVE NAME: PAT CHART RX NEW PRESCRIPTION
DESCRIPTION: Key necessary to access MEDS RX NEW option in the GUI PATIENT CHART program

NAME: BPCRXPMD
DESCRIPTIVE NAME: PAT CHART RX PCC MED DATE
DESCRIPTION: Key necessary to access MEDS RX PCC MED DATE option in the GUI PATIENT CHART program

NAME: BPCRXPMP
DESCRIPTIVE NAME: PAT CHART RX PCC MED PRINT
DESCRIPTION: Key necessary to access MEDS RX PCC MED PROFILE PRINT option in the GUI PATIENT CHART program

NAME: BPCRXPMR
DESCRIPTIVE NAME: PAT CHART RX PCC MED REFRESH
DESCRIPTION: Key necessary to access MEDS RX PCC MED REFRESH option in the GUI PATIENT CHART program

NAME: BPCRXR
DESCRIPTIVE NAME: PAT CHART RX REFILL
DESCRIPTION: Key necessary to access MEDS RX REFILL option in the GUI PATIENT CHART program

NAME: BPCSDP
DESCRIPTIVE NAME: PAT CHART APPT PRINT

DESCRIPTION: Key necessary to access APPT PRINT option in the GUI PAT CHART program.

NAME: BPCSPDA
DESCRIPTIVE NAME: PAT CHART SPECIALS ALLERGY
DESCRIPTION: Key necessary to access SPECIAL ALLERGY option in the GUI PATIENT CHART program

NAME: BPCSPDS
DESCRIPTIVE NAME: PAT CHART SPECIAL DIABETES
DESCRIPTION: Key necessary to access DIABETES SUMMARY option in the GUI PATIENT CHART program

NAME: BPCSPP
DESCRIPTIVE NAME: PAT CHART SPECIALS PRINT
DESCRIPTION: Key necessary to access SPECIALS PRINT option in the GUI PATIENT CHART program

NAME: BPCSPRC
DESCRIPTIVE NAME: PAT CHART SPECIAL REF CARE
DESCRIPTION: Key necessary to access SPECIAL REF CARE option in the GUI PATIENT CHART program

NAME: BPCSPTR
DESCRIPTIVE NAME: PAT CHART SPECIAL TROUBLESHOOT
DESCRIPTION: Key necessary to access TROUBLESHOOT option in the GUI PATIENT CHART program

NAME: BPCSPTRP
DESCRIPTIVE NAME: PAT CHART SPECIAL TRBSHT PRINT
DESCRIPTION: Key necessary to access TROUBLESHOOT PRINT option in the GUI PAT CHART program.

NAME: BPCTAP
DESCRIPTIVE NAME: PAT CHART TAB APPTS
DESCRIPTION: Key necessary to access APPT TAB option in the GUI PATIENT CHART program

NAME: BPCTCV
DESCRIPTIVE NAME: PAT CHART TAB COVER
DESCRIPTION: Key necessary to access COVER TAB option in the GUI PATIENT CHART program

NAME: BPCTEP
DESCRIPTIVE NAME: PAT CHART TAB EDUC PROTOCOL

DESCRIPTION: Key necessary to access EDUC PR TAB option in the GUI PATIENT CHART program

NAME: BPCTFS
DESCRIPTIVE NAME: PAT CHART TAB FACE SHEET
DESCRIPTION: Key necessary to access FACE TAB option in the GUI PATIENT CHART program

NAME: BPCTHS
DESCRIPTIVE NAME: PAT CHART TAB HLTH SUM
DESCRIPTION: Key necessary to access HLTH SUM TAB option in the GUI PATIENT CHART program

NAME: BPCTIM
DESCRIPTIVE NAME: PAT CHART TAB IMMUNIZATIONS
DESCRIPTION: Key necessary to access IMM TAB option in the GUI PATIENT CHART program

NAME: BPCTLR
DESCRIPTIVE NAME: PAT CHART TAB LAB
DESCRIPTION: Key necessary to access LAB TAB option in the GUI PATIENT CHART program

NAME: BPCTMS
DESCRIPTIVE NAME: PAT CHART TAB MEASUREMENT
DESCRIPTION: Key necessary to access MEAS TAB option in the GUI PATIENT CHART program

NAME: BPCTPC
DESCRIPTIVE NAME: PAT CHART TAB PCC OTHER
DESCRIPTION: Key necessary to access PCC TAB option in the GUI PATIENT CHART program

NAME: BPCTPR
DESCRIPTIVE NAME: PAT CHART TAB PROBLEM
DESCRIPTION: Key necessary to access PROBLEM TAB option in the GUI PATIENT CHART program

NAME: BPCTPV
DESCRIPTIVE NAME: PAT CHART TAB POV
DESCRIPTION: Key necessary to access POV TAB option in the GUI PATIENT CHART program

NAME: BPCTRG
DESCRIPTIVE NAME: PAT CHART TAB REG

DESCRIPTION: Key necessary to access REG TAB option in the GUI PATIENT CHART program

NAME: BPCTRX
DESCRIPTIVE NAME: PAT CHART TAB MEDICATIONS
DESCRIPTION: Key necessary to access MEDS TAB option in the GUI PATIENT CHART program

NAME: BPCTSP
DESCRIPTIVE NAME: PAT CHART TAB SPECIALS
DESCRIPTION: Key necessary to access SPECIALS TAB option in the GUI PATIENT CHART program

NAME: BPCTTN
DESCRIPTIVE NAME: PAT CHART TAB TELNET
DESCRIPTION: Key necessary to access TELNET TAB option in the GUI PATIENT CHART program

NAME: BPCTWH
DESCRIPTIVE NAME: PAT CHART TAB WOMEN HLTH
DESCRIPTION: Key necessary to access WOMEN H TAB option in the GUI PATIENT CHART program

NAME: BPCTXR
DESCRIPTIVE NAME: PAT CHART TAB XRAY
DESCRIPTION: Key necessary to access XRAY TAB option in the GUI PATIENT CHART program

NAME: BPCXRP
DESCRIPTIVE NAME: PAT CHART X-RAY PRINT
DESCRIPTION: Key necessary to access X-RAY PRINT option in the GUI PAT CHART program.

5.0 MENU DIAGRAM

The Patient Chart GUI Program does not contain any menus.

6.0 DATA DICTIONARIES

Table files and their appropriate dictionary field details are listed below.

FILE: REMOTE PROCEDURE
GLOBAL: ^XWB(8994,
FILE #: 8994

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	NAME	D0,0	1	F
.02	TAG	"	2	F
.03	ROUTINE	"	3	F
.04	RETURN VALUE TYPE	"	4	S
.05	AVAILABILITY	"	5	S
.06	INACTIVE	"	6	S
.07	CLIENT MANAGER	"	7	S
.08	WORD WRAP ON	"	8	S
1	DESCRIPTION (8994.01)			
.01	DESCRIPTION	D0,1,D1,0	1	W
2	INPUT PARAMETER (8994.02)			
.01	INPUT PARAMETER	D0,2,D1,0	1	F
.02	PARAMETER TYPE	"	2	S
.03	MAXIMUM DATA LENGTH	"	3	N
.04	REQUIRED	"	4	S
1	DESCRIPTION (8994.021)			
.01	DESCRIPTION	D0,2,D1,1,D2,0	1	W
3	RETURN PARAMETER DESCRIPTION (8994.03)			
.01	RETURN PARAMETER DESCRIPTION	D0,3,D1,0	1	W
9999999.01	RAW BUFFER IS INPUT PARAM	D0,9999999	1	S

FILE: BGU TRACE
GLOBAL: ^BGUTRACE(
FILE #: 90061

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	SEQUENCE NUMBER	D0,0	1	N
.02	APPLICATION ID	"		F
.03	$J	"	3	N
.04	DATE/TIME	"	4	D

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.05	BUFFER STRING (90061.01)			
.01	BUFFER STRING	D0,1,D1,0	1	W
.06	XQOR STRING (90061.02)			
.01	XQOR STRING	D0,2,D1,0	1	W
.07	RETURN DATA (90061.03)			
.01	RETURN DATA	D0,3,D1,0	1	W

FILE: BGU SITE PARAMETERS
GLOBAL: ^BGUSP(
FILE #: 90062

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	SITE	D0,0	1	P
.02	TRACE ALL	"	2	S
.03	$J TO TRACE (90062.01)			
.01	$J TO TRACE	D0,1,D1,0	1	N
1	Number of sessions to retain	"	2	N
.04	SOCKET	D0,2	1	N
.05	STATISTICS	D0,0	3	S
.06	SERVER TIMEOUT	"	6	N
.07	DAYS TO KEEP ENTRIES	"	7	N

FILE: BGU SIGN-ON STATISTICS
GLOBAL: ^BGUSEC(
FILE #: 90062.1

FIELD #	FIELDNAME	SUBSCRIPT	PIECE	TYPE
.001	DATE/TIME	D		
.01	USER	D0,0	1	P
1	DEVICE $I	"	2	F
2	JOB	"	3	N
3	SIGNOFF TIME	"	4	D
4	NODE NAME	"	5	F
5	SOCKET	"	7	F
6	ETHERNET ADDRESS	"	8	F
7	APPLICATION ID	"	6	F
10	PARAMETER	"	10	F

FIELD #	FIELDNAME	SUBSCRIPT	PIECE	TYPE
20	CALL STATISTICS (90062.11)			
.01	CALL STATISTICS	D0,20,D1,0	1	F
.02	NUMBER	"	2	N
97	ELAPSED TIME (SECONDS)	COMPUTED		
99	ELAPSED TIME (MINUTES	"		

FILE: BGU SERVER
GLOBAL: ^BGUSV(
FILE #: 90063

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	NAME	D0,0	1	F
1	PORT	"	2	F

FILE: BGU TABLE
GLOBAL: ^BGUT(
FILE #: 90064

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	NAME	D0,0	1	F
1	FILE NUMBER	"	2	F
2	FIELD (90064.02)			
.01	FIELD	D0,1,D1,0	1	F
1	FIELD NUMBER	"	2	F

FILE: BGU EXECUTE CODE
GLOBAL: ^BGUEXC(
FILE #: 90064.1

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	NAME	D0,0	1	F
1	EXECUTE CODE	D0,.1	E1,220	F
2	ACTIVATION	D0,0	2	S
3	TYPE	"	3	S
4	ERROR PROMPT	D0,.2	1	F
5	APPLICATION NAME	D0,2	1	P
6	DESCRIPTION (90064.101)			
.01	DESCRIPTION	D0,1,D1,0	1	W

FILE: BGU GENLIST TEMPLATE

GLOBAL: ^BGUPLATE(
FILE #: 90064.2

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	NAME	D0,0	1	F
.02	BGUFILE	"	2	P
.03	BGUIEN	"	3	N
.04	BGUMORE	"	4	F
.05	BGUCRFS	"	5	F
.06	BGUMAX	"	6	N
.07	BGUBEGIN	"	7	F
.08	BGUEND	"	8	F
.09	BGUDIR	"	9	F
.1	BGUID	"	10	F
1	BGUVLST (90064.21)			
.01	BGUVLST	D0,1,D1,0	1	W
2	BGUSCR	D0,2	1	F
3	BGUCNDS	D0,3	1	F
4	REMOTE CALL	D0,4	1	F

FILE: BGU MUMPS CODE
GLOBAL: ^BGUMCD(
FILE #: 90065

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	NAME	D0,0	1	F
.02	MUMPS CODE	"	2	F
.03	DATE	"	3	D
.04	ACTIVE FLAG	"	4	S
.05	DEVELOPER	"	5	P
.06	DESCRIPTION (90065.01)			
.01	DESCRIPTION	D0,1,D1,0	1	W

FILE: BGU PRESET REMOTE PROCEDURE CALL
GLOBAL: ^BGUPSRPC(
FILE #: 90066

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	NAME	D0,0	1	F
.02	REMOTE PROCEDURE	"	2	P
.03	PARAMETER (90066.01)			
.01	PARAMETER NUMBER	D0,1,D1,0	1	N

.02	VALUE	"	2	F

FILE: BGU USER GUI PROFILE
GLOBAL: ^BGUUSER(
FILE #: 90069

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.01	NAME	D0,0	1	P
21	INSTITUTION (90069.021)			
.01	INSTITUTION	D0,21,D1,0	1	P
.02	INBOUND CONNECTION ALLOWED	"	2	S
.03	OUBOUND CONNECTION ALLOWED	"	3	S
.04	IP ADDRESS	"	4	F
.05	IP PORT NUMBER (SOCKET)	"	5	N
.06	ACCESS CODE	"	6	F
.07	VERIFY CODE	"	7	F
11	ALLOWED MAC ADDRESS (90069.2111)			
.01	ALLOWED MAC ADDRESS	D0,21,D1,11,D2,0	1	F
21	BGU PACKAGE SUPPORTED (90069.2121)			
.01	BGU PACKAGE SUPPORTED	D0,21,D1,21,D2,0	1	P
.02	PARAM02	"	2	F
.03	PARAM03	"	3	F
.04	PARAM04	"	4	F
.05	PARAM05	"	5	F
.06	PARAM06	"	6	F
.07	PARAM07	"	7	F
.08	PARAM08	"	8	F
51	PACKAGE (90069.051)			
.01	PACKAGE	D0,51,D1,0	1 P	
.02	PARAM02	"	2	F
.03	PARAM03	"	3	F
.04	PARAM04	"	4	F
.05	PARAM05	"	5	F
.06	PARAM06	"	6	F
.07	PARAM07	"	7	F

FIELD #	FIELD NAME	SUBSCRIPT	PIECE	TYPE
.08	PARAM08	"	8	F

7.0 EXTERNAL RELATIONS

The Patient Chart application references several packages. When possible, external published entry points are called to obtain data or data is obtained using the MDAO interface referenced in Appendix A. When updating RPMS data, either the published entry points or fileman edit calls are used. There are no formal agreements with other applications.

8.0 ENTRY POINTS

8.1 Remote Procedure Captioned Listing

The following table lists the remote procedures used by the Patient Chart and the associated settings used in the remote procedure. These are considered the entry points to the RPMS system.

```
NAME: BGU ACCESSCODE CHANGE        TAG: CHGACODE
  ROUTINE: BGUXUSRC                RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU APICALL                  TAG: APICALL
  ROUTINE: BGUAPI                  RETURN VALUE TYPE: SINGLE VALUE
 DESCRIPTION:
 This takes care of an APICALL call from the client

NAME: BGU APITABLE                 TAG: APICALL
  ROUTINE: BGUAPI                  RETURN VALUE TYPE: ARRAY
 DESCRIPTION:
 This takes care of an APITABLE call from the client

NAME: BGU AVLOGON                  TAG: VALIDAV
  ROUTINE: BGUXUSRB                RETURN VALUE TYPE: ARRAY
 DESCRIPTION:
 This call is utilized to verify the access and verify codes  passed by the client and to set the
environment.
INPUT PARAMETER: ACODE             PARAMETER TYPE: SINGLE VALUE
 DESCRIPTION:
 Access code encrypted from client
INPUT PARAMETER: VCODE             PARAMETER TYPE: SINGLE VALUE
 DESCRIPTION:
 Verify code encrypted from client
INPUT PARAMETER: BGUETH            PARAMETER TYPE: SINGLE VALUE
 DESCRIPTION:
 Ethernet hex address of client card
INPUT PARAMETER: BGUAPP            PARAMETER TYPE: SINGLE VALUE
 DESCRIPTION:
 Application ID/Version
INPUT PARAMETER: BGUPRM            PARAMETER TYPE: SINGLE VALUE
 DESCRIPTION:
 Security parameter passed by client
INPUT PARAMETER: TEST              PARAMETER TYPE: SINGLE VALUE
 DESCRIPTION:
 THIS DESCRIBES THE TEST PARAMETER LINE 1
 LINE 2 FOR TEST PARAMETER
 LINE 3 FOR TEST PARAMETER

NAME: BGU CAC                          TAG: CAC
  ROUTINE: BGUXUSRB                    RETURN VALUE TYPE: ARRAY
 DESCRIPTION:
 This call is utilized to change the ACCESS code for a user on the databaseserver

NAME: BGU CFAC                         TAG: CFAC
  ROUTINE: BGUXUSRB                    RETURN VALUE TYPE: ARRAY
 DESCRIPTION:
 This call is utilized to change the facility ID by the client

NAME: BGU CREATERECOR                  TAG: CREATE
  ROUTINE: BGULIB                      RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU CVC                          TAG: CVC
  ROUTINE: BGUXUSRB                    RETURN VALUE TYPE: ARRAY
 DESCRIPTION:
 This call is utilized to change the VERIFY code for a user on the database server.

NAME: BGU DESTRUCTOR                   TAG: EN
  ROUTINE: BGUDSTR                     RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC
 DESCRIPTION:
 Called by the MDAO to kill ^TMP("SAVE",BGUID) to clear state between when the file object
is terminated, for executables or a Web page.

NAME: BGU FILER                        TAG: EN
  ROUTINE: BGUFLR                      RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU GENLIST                      TAG: EN
  ROUTINE: BGULIST                     RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU GETDE                        TAG: EN
  ROUTINE: BGUGDE                      RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU GETPATIENTLIST               TAG: PTLIST
  ROUTINE: BGUGPLK                     RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU PMR                          TAG: EN
```

```
  ROUTINE: BGUPMR                 RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU PSRPC                   TAG: EN
  ROUTINE: BGUPSRPC               RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU ROUTINE FILER           TAG: EN
  ROUTINE: BGURTNFL               RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC            RAW BUFFER IS INPUT PARAM: TRUE

NAME: BGU RPCCALL                 TAG: RPCCALL
  ROUTINE: BGUAPI                 RETURN VALUE TYPE: SINGLE VALUE

NAME: BGU RPCTABLE                TAG: RPCCALL
  ROUTINE: BGUAPI                 RETURN VALUE TYPE: ARRAY

NAME: BGU SELRTNS                 TAG: SEL
  ROUTINE: BGUPMR                 RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU TABLENAMES              TAG: EN
  ROUTINE: BGUFILE                RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BGU VERIFYCODE CHANGE       TAG: CHGVCODE
  ROUTINE: BGUXUSRC               RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC CHARTDATA               TAG: BPCCHT
  ROUTINE: BPCPC                  RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC EDITNOTE                TAG: NOTEDIT
  ROUTINE: BPCNOTEC               RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC
 DESCRIPTION:
 Call to edit problem note data

NAME: BPC EDITPROBLEM             TAG: PRBEDIT
  ROUTINE: BPCPRBC                RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC
 DESCRIPTION:
 Call used for adding,editing, and deleting problems

NAME: BPC EDITREFERRAL            TAG: REFEDIT
```

```
 ROUTINE: BPCRC3                     RETURN VALUE TYPE: ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GET IMM FORECAST DATA          TAG: GETIFDSP
 ROUTINE: BPCIMDSP                   RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GET IMM HISTORY DATA  TAG: GETIHDSP
 ROUTINE: BPCIMDSP                   RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GET INITIAL LAB LIST       TAG: TAG
 ROUTINE: BPCLIPL                    RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC
 DESCRIPTION:
 This collects the initial lab Pick List for a provider and returns the names of:
          1. patients from the lab edit/disp opt
          2. returns patients who have labs today and these are flagged with *L*
          3. returns lab data for flagged patients.

NAME: BPC GET RX NEXT NUMBER     TAG: GETDEA
 ROUTINE: BPCRXNXT                   RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GET SD APPT DATA           TAG: GETAPPT
 ROUTINE: BPCAPPT                    RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GETAGENTS                  TAG: AGENTS
 ROUTINE: BPCPC2                     RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC
 DESCRIPTION:
 RETURNS ALLERGIC CAUSITIVE AGENT(S)
INPUT PARAMETER: BPCAGENT            PARAMETER TYPE: SINGLE VALUE
 MAXIMUM DATA LENGTH: 255            REQUIRED: NO
 DESCRIPTION:
 THE CAUSATIVE AGENT FOR WHICH TO RETURN DATA. IF THIS PARAMETER
BEGINS WITH A "?", THE CALL RETURNS A LIST OF DATA WHICH STARTS WITH
THE STRING AFTER  THE "?". OTHERWISE, THE CALL RETURNS DATA FOR A
DIRECT HIT IF ANY.

NAME: BPC GETALLERG                  TAG: GETALRG
 ROUTINE: BPCALRG                    RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GETBENEFICIARIES           TAG: BENELIST
```

```
  ROUTINE: BPCPC1                     RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETBWDATA                   TAG: GETBWDSP
  ROUTINE: BPCBWDSP                   RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETCLINLIST                 TAG: CLNLIST
  ROUTINE: BPCRC1                     RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETCOMMUNITIES              TAG: COMLIST
  ROUTINE: BPCPC1                     RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETCPTCATS                  TAG: CPTLIST
  ROUTINE: BPCRC                      RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETCPTLIST                  TAG: PXLIST
  ROUTINE: BPCRC4                     RETURN VALUE TYPE: P

NAME: BPC GETDBS                      TAG: DBSUM
  ROUTINE: BPCDBS                     RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETDIAGNOSISLIST            TAG: DXLIST
  ROUTINE: BPCGDX                     RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC
 DESCRIPTION:
 Gets list of diagnosises for client apps

NAME: BPC GETDRGLIST                  TAG: DRGLIST
  ROUTINE: BPCRC5                     RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETDXLIST                   TAG: DXLIST
  ROUTINE: BPCRC4                     RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETFACEDATA                 TAG: GETFACE
  ROUTINE: BPCRC8                     RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETHSDATA                   TAG: GETHSDSP
  ROUTINE: BPCHSDSP                   RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETICDCATS               TAG: ICDLIST
  ROUTINE: BPCRC                    RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETINTERIMLABS           TAG: INT
  ROUTINE: BPCLAB                   RETURN VALUE TYPE: GLOBAL ARRAY
  WORD WRAP ON: TRUE

NAME: BPC GETLOCALSERVICECAT       TAG: SCATLIST
  ROUTINE: BPCRC2                   RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETLOCATIONS             TAG: LOCLIST
  ROUTINE: BPCRC                    RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETMYLABS                TAG: GETMLDSP
  ROUTINE: BPCLRML                  RETURN VALUE TYPE: GLOBAL ARRAY
  WORD WRAP ON: TRUE

NAME: BPC GETPICKLIST              TAG: GETPICK
  ROUTINE: BPCLORD                  RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETPROBLEMS              TAG: EN
  ROUTINE: BPCPROB                  RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETPROVIDERS             TAG: PRVLIST
  ROUTINE: BPCRC                    RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETRCISPROV              TAG: RPRVLIST
  ROUTINE: BPCRC1                   RETURN VALUE TYPE: GLOBAL ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETRCSITEPARAM           TAG: RCISPARM
  ROUTINE: BPCRC2                   RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC

NAME: BPC GETREFERRALDATA          TAG: GETREF
  ROUTINE: BPCRC5                   RETURN VALUE TYPE: ARRAY
  AVAILABILITY: PUBLIC
```

```
NAME: BPC GETREFERRALDEF           TAG: GETTEMP
 ROUTINE: BPCRC7                   RETURN VALUE TYPE: ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GETREFERRALS             TAG: REFLIST
 ROUTINE: BPCRC1                   RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GETREFLETTER             TAG: GETLET
 ROUTINE: BPCRC9                   RETURN VALUE TYPE: ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GETRELIGIONS             TAG: RELLIST
 ROUTINE: BPCPC2                   RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GETSTATES                TAG: STLIST
 ROUTINE: BPCPC1                   RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GETTEMPLATELIST          TAG: TEMPLIST
 ROUTINE: BPCRC7                   RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GETTRIBES                TAG: TRIBLIST
 ROUTINE: BPCPC1                   RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC GETVENDORS               TAG: VNDLIST
 ROUTINE: BPCRC2                   RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC

NAME: BPC IMM LOT REQUIRED         TAG: USELOT
 ROUTINE: BPCIMDSP                 RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC
DESCRIPTION:
Gets flag to determine for a given Immunization if lot numbers are required.

NAME: BPC MEASTYPE HELP            TAG: HELP
 ROUTINE: BPCMEAS                  RETURN VALUE TYPE: GLOBAL ARRAY
 AVAILABILITY: PUBLIC
DESCRIPTION:
CALLS HELP STRINGS FROM APCDVMSR ROUTINE FOR MEAS TYPE
RETURN PARAMETER DESCRIPTION:
RETURNS STRING OF HELP INFO OR ERROR
```

NAME: BPC MEASVAL VALIDATE TAG: VAL
ROUTINE: BPCMEAS RETURN VALUE TYPE: GLOBAL ARRAY
AVAILABILITY: PUBLIC
DESCRIPTION:
validates measurement value to APCDVMS2 routine
RETURN PARAMETER DESCRIPTION:
returns error code or 'INPUT TRANSFORM OK!'

NAME: BPC MEASVISIT SAVE TAG: GETVISIT
ROUTINE: BPCMEAS RETURN VALUE TYPE: GLOBAL ARRAY
AVAILABILITY: PUBLIC
DESCRIPTION:
creates a visit and measurement for the patient.
called from Patient Chart GUI application Add Measurment form
RETURN PARAMETER DESCRIPTION:
returns error message or 'PCC Measurement Saved"

NAME: BPC ORDER LABS TAG: START
ROUTINE: LRORD RETURN VALUE TYPE: ARRAY
AVAILABILITY: PUBLIC
DESCRIPTION:
Allows user to place a lab order if LAB(5.1) is installed

NAME: BPC ORDERSCREEN TAG: LISTT
ROUTINE: BPCLORD RETURN VALUE TYPE: ARRAY
AVAILABILITY: PUBLIC

NAME: BPC PRINT LAB ORDER TAG: PRTORD
ROUTINE: BPCLOPRT RETURN VALUE TYPE: GLOBAL ARRAY
AVAILABILITY: PUBLIC

NAME: BPC RX EXPAND SIG TAG: GETSIG
ROUTINE: BPCRXSIG RETURN VALUE TYPE: GLOBAL ARRAY
AVAILABILITY: PUBLIC

9.0 EXPORTED OPTIONS

9.1 Purge The Gui Sign On Log

NAME: BGU SITE PARAMETER ENTER/EDIT
MENU TEXT: RPC Broker Site Parameter Enter/Edit
TYPE: edit
CREATOR: EVANS,F.J.
DESCRIPTION: This will set trace all for all jobs, or you can specify a $J for it to trace on.
DIC {DIC}: BGUSP(DIC(0): AEMQL
DIE: BGUSP(DR {DIE}: .01:2
UPPERCASE MENU TEXT: RPC BROKER SITE PARAMETER ENTE

NAME: BGU START LISTENER
MENU TEXT: Start Listener
TYPE: run routine
CREATOR: EVANS,F.J.
DESCRIPTION: This option starts the background listener for communication with other computers
ROUTINE: STRT^BGUTCP(8000)
UPPERCASE MENU TEXT: START LISTENER

NAME: BGU START TRACE
MENU TEXT: Start Trace
TYPE: run routine
CREATOR: EVANS,F.J.
LOCK: BGUZMENU
DESCRIPTION: Start Trace for all jobs or for a specific $J.
ROUTINE: START^BGUTUTL
UPPERCASE MENU TEXT: START TRACE

NAME: BGU STOP LISTENER
MENU TEXT: Stop Listener
TYPE: run routine
CREATOR: EVANS,F.J.
DESCRIPTION: This option stops the background listener for communicating with other computers
ROUTINE: STOP^BGUTCP(8000)
UPPERCASE MENU TEXT: STOP LISTENER

NAME: BGU STOP TRACE
MENU TEXT: Stop Trace
TYPE: run routine
CREATOR: EVANS,F.J.
LOCK: BGUZMENU
DESCRIPTION: Stop Trace for all jobs or for a specific $J.
ROUTINE: STOP^BGUTUTL
UPPERCASE MENU TEXT: STOP TRACE

NAME: BGU SYSTEM MANAGER MENU
MENU TEXT: IHS RPC Broker System Manager Options
TYPE: menu
CREATOR: EVANS,F.J.
PACKAGE: IHS RPC BROKER
DESCRIPTION: This is the option menu for the System Manager of the IHS RPC Broker Package
ITEM: BGU START LISTENER
SYNONYM: STR
DISPLAY ORDER: 5

ITEM: BGU STOP LISTENER
SYNONYM: STP
DISPLAY ORDER: 10

ITEM: BGU SITE PARAMETER ENTER/EDIT
SYNONYM: EDT
DISPLAY ORDER: 15
TIMESTAMP: 58665,36108
UPPERCASE MENU TEXT: IHS RPC BROKER SYSTEM MANAGER

NAME: BGU TRACE CLEANUP
MENU TEXT: Clean Out Old Trace Sessions
TYPE: action
CREATOR: EVANS,F.J.
LOCK: BGUZMENU
E ACTION PRESENT: YES
ENTRY ACTION: D BJENT^BGUTRCU
UPPERCASE MENU TEXT:CLEAN OUT OLD TRACE SESSIONS

NAME: BGU TRACE INQUIRY
MENU TEXT: Display Trace Session Data
TYPE: run routine
CREATOR: EVANS,F.J.

LOCK: BGUZMENU
DESCRIPTION: Displays the last sequence number recorded in the BGU Trace file and allows the user to display any range of sequence numbers in the file.
After the user presses enter to the Start prompt without data another prompt requests the BGU Trace Sequence number to display. If sequence numbers are entered data is displayed in the same manner as would be displayed from Inquiry mode in P^DI.
ROUTINE: BGUTRLU
UPPERCASE MENU TEXT: DISPLAY TRACE SESSION DATA

NAME: BGUMENU
MENU TEXT: IHS RPC Main Menu
TYPE: menu
CREATOR: EVANS,F.J.
LOCK: BGUZMENU
PACKAGE: IHS RPC BROKER
DESCRIPTION: This is the main menu option for the IHS RPC Broker
ITEM: BGU START LISTENER
  SYNONYM: SM
  DISPLAY ORDER: 1
ITEM: BGU STOP LISTENER
  SYNONYM: STP
  DISPLAY ORDER: 2
ITEM: BGU SITE PARAMETER ENTER/EDIT
  SYNONYM: SP
  DISPLAY ORDER: 4
ITEM: BGU START TRACE
  SYNONYM: STT
  DISPLAY ORDER: 4
ITEM: BGU STOP TRACE
  SYNONYM: SPT
  DISPLAY ORDER: 5
ITEM: BGU TRACE CLEANUP
  SYNONYM: CLN
  DISPLAY ORDER: 6
ITEM: BGU TRACE INQUIRY
  SYNONYM: TRAC
  DISPLAY ORDER: 7
Short Menu Text: Clean Out Old Trace Sessions
TIMESTAMP: 58665,36108
UPPERCASE MENU TEXT: IHS RPC MAIN MENU

NAME: BGUTASK TRACE ENTRY CLEANUP

MENU TEXT: TASK THE BGU TRACE ENTRY CLEANUP NIGHTLY
TYPE: run routine
CREATOR: EVANS,F.J.
LOCK: BGUZMENU
DESCRIPTION: This task will purge entries from the BGU TRACE file, based on the DAYS TO KEEP ENTRIES field in the BGU SITE PARAMETER file, and using the DATE/TIME field in the ^BGUTRACE field as the reference for each entry.
Short Menu Text: BGU TRACE FILE CLEANUP
ROUTINE: BGUCU
UPPERCASE MENU TEXT: TASK THE BGU TRACE ENTRY CLEAN

10.0 Callable Routines

10.1 Remote Procedure Tag and Routine Listing

The following table lists the remote procedures used by the Patient Chart and the associated TAG and ROUTINE that is called by the remote procedure. These tags and routines are considered callable entry points but are only available thru the GUI interface. Message data can be viewed by using the trace option referenced in other parts of this document.

10.2 Remote Procedure List

NAME	TAG	ROUTINE
BGU ACCESSCODE CHANGE	CHGACODE	BGUXUSRC
BGU APICALL	APICALL	BGUAPI
BGU APITABLE	APICALL	BGUAPI
BGU AVLOGON	VALIDAV	BGUXUSRB
BGU CAC	CAC	BGUXUSRB
BGU CFAC	CFAC	BGUXUSRB
BGU CREATERECORD	CREATE	BGULIB
BGU CVC	CVC	BGUXUSRB
BGU DESTRUCTOR	EN	BGUDSTR
BGU FILER	EN	BGUFLR
BGU GENLIST	EN	BGULIST
BGU GETDE	EN	BGUGDE
BGU GETPATIENTLIST	PTLIST	BGUGPLK
BGU PMR	EN	BGUPMR
BGU PSRPC	EN	BGUPSRPC
BGU ROUTINE FILER	EN	BGURTNFL
BGU RPCCALL	RPCCALL	BGUAPI
BGU RPCTABLE	RPCCALL	BGUAPI
BGU SELRTNS	SEL	BGUPMR
BGU TABLENAMES	EN	BGUFILE
BGU VERIFYCODE CHANGE	CHGVCODE	BGUXUSRC
BPC CHARTDATA	BPCCHT	BPCPC
BPC EDITNOTE	NOTEDIT	BPCNOTEC
BPC EDITPROBLEM	PRBEDIT	BPCPRBC
BPC EDITREFERRAL	REFEDIT	BPCRC3
BPC GET IMM FORECAST DATA	GETIFDSP	BPCIMDSP
BPC GET IMM HISTORY DATA	GETIHDSP	BPCIMDSP
BPC GET INITIAL LAB LIST	TAG	BPCLIPL
BPC GET RX NEXT NUMBER	GETDEA	BPCRXNXT
BPC GET SD APPT DATA	GETAPPT	BPCAPPT
BPC GETAGENTS	AGENTS	BPCPC2
BPC GETALLERG	GETALRG	BPCALRG

BPC GETBENEFICIARIES	BENELIST	BPCPC1
BPC GETBWDATA	GETBWDSP	BPCBWDSP
BPC GETCLINLIST	CLNLIST	BPCRC1
BPC GETCOMMUNITIES	COMLIST	BPCPC1
BPC GETCPTCATS	CPTLIST	BPCRC
BPC GETCPTLIST	PXLIST	BPCRC4
BPC GETDBS	DBSUM	BPCDBS
BPC GETDIAGNOSISLIST	DXLIST	BPCGDX
BPC GETDRGLIST	DRGLIST	BPCRC5
BPC GETDXLIST	DXLIST	BPCRC4
BPC GETFACEDATA	GETFACE	BPCRC8
BPC GETHSDATA	GETHSDSP	BPCHSDSP
BPC GETICDCATS	ICDLIST	BPCRC
BPC GETINTERIMLABS	INT	BPCLAB
BPC GETLOCALSERVICECAT	SCATLIST	BPCRC2
BPC GETLOCATIONS	LOCLIST	BPCRC
BPC GETMYLABS	GETMLDSP	BPCLRML
BPC GETPICKLIST	GETPICK	BPCLORD
BPC GETPROBLEMS	EN	BPCPROB
BPC GETPROVIDERS	PRVLIST	BPCRC
BPC GETRCISPROV	RPRVLIST	BPCRC1
BPC GETRCSITEPARAM	RCISPARM	BPCRC2
BPC GETREFERRALDATA	GETREF	BPCRC5
BPC GETREFERRALDEF	GETTEMP	BPCRC7
BPC GETREFERRALS	REFLIST	BPCRC1
BPC GETREFLETTER	GETLET	BPCRC9
BPC GETRELIGIONS	RELLIST	BPCPC2
BPC GETSTATES	STLIST	BPCPC1
BPC GETTEMPLATELIST	TEMPLIST	BPCRC7
BPC GETTRIBES	TRIBLIST	BPCPC1
BPC GETVENDORS	VNDLIST	BPCRC2
BPC IMM LOT REQUIRED	USELOT	BPCIMDSP
BPC MEASTYPE HELP	HELP	BPCMEAS
BPC MEASVAL VALIDATE	VAL	BPCMEAS
BPC MEASVISIT SAVE	GETVISIT	BPCMEAS
BPC ORDER LABS	START	LRORD
BPC ORDERSCREEN	LISTT	BPCLORD
BPC PRINT LAB ORDER	PRTORD	BPCLOPRT
BPC RX EXPAND SIG	GETSIG	BPCRXSIG

11.0 CROSS REFERENCES

There are no Cross References in the Patient Chart GUI Program.

12.0 INTERNAL RELATIONS

The MDAO.dll file is an in process Active-X Automation Server used by the Visual Basic program that references it. This automation server exposes two objects: the CServer object and the CMFile object. The CServer object contains methods and properties which pertain to managing a connection to a MUMPS database server. The CMFile object works in conjunction with the CServer object . It contains methods and properties which allow the VB programmer to easily fetch, add, change, or delete data in a FileMan based MUMPS file on the MUMPS server. See Appendix A for details.

13.0 ARCHIVING AND PURGING

13.1 Purge of the Security Signon Log

Every user who signs on to RPMS using the GUI is registered. Infrequently the site manager may want to purge old entries to refresh space on the system. Purge the GUI Sign On Log option on the BGU Main Menu allows the site manager to purge old entries as needed.

```
IHS RPC (BGU) VERSION 1.2 MAIN MENU     DEMO HOSPITAL


   SM     Start Listener
   STP    Stop Listener
   SP     RPC Broker Site Parameter Enter/Edit
   STT    Start Trace
   SPT    Stop Trace
   CLN    Clean Out Old Trace Sessions
   PURG   Purge the GUI Sign On Log
   DSP    Display Trace Session Data


Select IHS RPC Main Menu Option: PURG  Purge the GUI Sign On Log

Purge BGU Sign-on log older than 30 days?? YES
BGU Sign-on log has been purged.



   SM     Start Listener
   STP    Stop Listener
   SP     RPC Broker Site Parameter Enter/Edit
   STT    Start Trace
   SPT    Stop Trace
   CLN    Clean Out Old Trace Sessions
   PURG   Purge the GUI Sign On Log
   DSP    Display Trace Session Data
```

13.2 Clean Out Old Trace Sessions

```
IHS RPC (BGU) VERSION 1.2 MAIN MENU     DEMO HOSPITAL


   SM     Start Listener
   STP    Stop Listener
   SP     RPC Broker Site Parameter Enter/Edit
   STT    Start Trace
   SPT    Stop Trace
   CLN    Clean Out Old Trace Sessions
   PURG   Purge the GUI Sign On Log
   DSP    Display Trace Session Data

Select IHS RPC Main Menu Option: CLN  Clean Out Old Trace Sessions


   SM     Start Listener
   STP    Stop Listener
   SP     RPC Broker Site Parameter Enter/Edit
   STT    Start Trace
   SPT    Stop Trace
   CLN    Clean Out Old Trace Sessions
   PURG   Purge the GUI Sign On Log
   DSP    Display Trace Session Data
Select IHS RPC Main Menu Option:

The option: BGUTASK TRACE ENTRY CLEANUP can also be run to setup the daily
scheduled task to clean out the old trace data.

>D ^XUP

Setting up programmer environment
Terminal Type set to: C-VT100

You have 22 new messages.
Select OPTION NAME: BGUTASK TRACE ENTRY CLEANUP          TASK THE BGU TRACE
ENTR
Y CLEANUP NIGHTLY
TASK THE BGU TRACE ENTRY CLEANUP NIGHTLY
```

14.0 GENERATING ONLINE DOCUMENTATION

Throughout the entire Patient Chart Program, the user may press the F1 key to access an online help file. This file allows the user to search for topics by index or by using the full-text search function.

Where data enry is required, users can put a single question mark in the choose option and get a display. For example, when selecting locations or providers.

There are no special templates in this package designed for online documentation.

15.0 Editing the RPC Broker Site Parameter File

The following example demonstrates the process to change site parameters used in the BGU namespace.

Select BGU SITE PARAMETERS: determines the facility location. There should be only one entry until future remote data views are enabled. Choose your facility name.
SITE: DEMO-HO.BIL.IHS.GOV//

TRACE ALL: Allows user to turn on trace function for all users. This allows the system to maintain a complete audit of all activity occurring through the GUI interface. When turned on, the interface will be slightly slower to the user. See the detailed use of this action within this manual.

Select $J TO TRACE: A specific partition can be monitored with this option selected. Generally, the site manage should just activate the TRACE ALL entry and not use this field.

SOCKET: 8000// Allows the site manager to change the listening port. The default for IHS is 8000.

STATISTICS: NO// Allows user to monitor GUI sign-on and number and types of messages exercised. When turned on, individual remote procedure use by user can be viewed. See example in this manual.

SERVER TIMEOUT: 3600// This determines when a server partition should shutdown from no activity with the client.

DAYS TO KEEP ENTRIES: 10// This determines the number of days to keep TRACE DATA when the daily task to cleanout the trace file is activated. See example in this manual.

Example:

```
Select BGU SITE PARAMETERS:
IHS RPC (BGU) VERSION 1.2 MAIN MENU     DEMO HOSPITAL


   SM     Start Listener
   STP    Stop Listener
   SP     RPC Broker Site Parameter Enter/Edit
   STT    Start Trace
   SPT    Stop Trace
   CLN    Clean Out Old Trace Sessions
   PURG   Purge the GUI Sign On Log
   DSP    Display Trace Session Data


Select IHS RPC Main Menu Option: SP  RPC Broker Site Parameter Enter/Edit
Select BGU SITE PARAMETERS:    DEMO-HO.BIL.IHS.GOV
SITE: DEMO-HO.BIL.IHS.GOV//
TRACE ALL: YES// ??
    Allows user to turn on trace function for all users
    Choose from:
      Y        YES
      N        NO
TRACE ALL: YES//
Select $J TO TRACE:
SOCKET: 8000//
STATISTICS: NO// ??
    Allows user to monitor GUI sign-on and number and types
    of messages exercised.
    Choose from:
      Y        YES
      N        NO
STATISTICS: NO// Y  YES
SERVER TIMEOUT: 3600//
DAYS TO KEEP ENTRIES: 10//
```

16.0 Security Sign On Options with Patient Chart

As with the regular RPMS Kernel, the Patient Chart supports a log capturing user sign on, and sign off activity. Data can be viewed using FileMan. The example below shows the activity for an individual session for a user complete with number of calls made and where the call originated.

```
D P^DI

Select OPTION: INQUIRE TO FILE ENTRIES

OUTPUT FROM WHAT FILE: BGU SIGN-ON STATISTICS//
Select BGU SIGN-ON STATISTICS USER: EVANS,F.J.
ANOTHER ONE:
STANDARD CAPTIONED OUTPUT? Yes//   (Yes)
Include COMPUTED fields:  (N/Y/R/B): NO//  - No record number (IEN), no
Computed
 Fields

DATE/TIME: OCT 01, 2001@14:23:32        USER: EVANS,F.J.
  DEVICE $I: 56                         JOB: 29
  SIGNOFF TIME: OCT 01, 2001@14:23:58   NODE NAME: 161.223.9.200
  SOCKET: 4911                          ETHERNET ADDRESS: 161.223.9.200
CALL STATISTICS: BGU AVLOGON            NUMBER: 1
CALL STATISTICS: BGU CFAC               NUMBER: 1
CALL STATISTICS: BGU DESTRUCTOR         NUMBER: 27
CALL STATISTICS: BGU GENLIST            NUMBER: 17
CALL STATISTICS: BGU GETDE              NUMBER: 17
CALL STATISTICS: BPC CHARTDATA          NUMBER: 1
CALL STATISTICS: BPC GET IMM FORECAST DATA
  NUMBER: 1
CALL STATISTICS: BPC GET IMM HISTORY DATA
  NUMBER: 1
CALL STATISTICS: BPC GET SD APPT DATA   NUMBER: 1
CALL STATISTICS: BPC GETFACEDATA        NUMBER: 1
CALL STATISTICS: BPC GETPROBLEMS        NUMBER: 1
```

17.0 Auditing with Patient Chart

17.1 Displaying a Trace Session

The following is an example of examining a trace (audit) session. In the 69 entries below, the user signs on to the system and establishes a session by entering a good hashed access and verify code. (entry 1) Package application information is obtained (4,7,10,13,16) using MDAO, security keys for the user are obtained (20,24) using MDAO, selecting a patient (27,28) using MDAO and then populating the tabs with patient data. Finally actual data is displayed for an individual tracing.

```
IHS RPC (BGU) VERSION 1.2 MAIN MENU     DEMO HOSPITAL


   SM     Start Listener
   STP    Stop Listener
   SP     RPC Broker Site Parameter Enter/Edit
   STT    Start Trace
   SPT    Stop Trace
   CLN    Clean Out Old Trace Sessions
   PURG   Purge the GUI Sign On Log
   DSP    Display Trace Session Data

Select IHS RPC Main Menu Option: DSP  Display Trace Session DataLAST SEQUENCE
AS
SIGNED: 69
START FROM:  1
END AT: 69
1 BGU AVLOGON^K2SFn|l1dBS(JU0$5irx^k&\>H]QEyVgp?!lVFYw^161.223.9.200^^
2 BGU CFAC^2248
3 BGU GETDE^PACKAGE^13
4 BGU GENLIST^PACKAGE^^1^B^25^OUTPATIENT PHARMACY^OUTPATIENT
PHARMACY^13^^^11423
33374^
5 BGU DESTRUCTOR^1142333374
6 BGU GETDE^PACKAGE^13
7 BGU GENLIST^PACKAGE^^1^B^25^IHS LABORATORY^IHS LABORATORY^13^^^1142333374^
8 BGU DESTRUCTOR^1142333374
9 BGU GETDE^PACKAGE^13
10 BGU GENLIST^PACKAGE^^1^B^25^IMMUNIZATION^IMMUNIZATION^13^^^1142333374^
11 BGU DESTRUCTOR^1142333374
12 BGU GETDE^PACKAGE^13
13 BGU GENLIST^PACKAGE^^1^B^25^REFERRED CARE INFO SYSTEM^REFERRED CARE INFO
SYST
EM^13^^^1142333374^
14 BGU DESTRUCTOR^1142333374
15 BGU GETDE^PACKAGE^13
16 BGU GENLIST^PACKAGE^^1^C^25^GMRA^GMRA^13^^^1142333374^
17 BGU DESTRUCTOR^1142333374
18 BGU DESTRUCTOR^1142333374
19 BGU GETDE^NEW PERSON^3,51!.01
20 BGU GENLIST^NEW PERSON^2^^^^^^3,51!.01^^^1142334996^51!.01 .PM. 1"BPC".E
<>
21 BGU DESTRUCTOR^1142334996
22 BGU DESTRUCTOR^1142334996
23 BGU GETDE^NEW PERSON^3,51!.01
```

```
24 BGU GENLIST^NEW PERSON^2^^^^^^3,51!.01^^^1142335667^51!.01 .PM. 1"BGU".E
25 BGU DESTRUCTOR^1142335667
26 BGU DESTRUCTOR^1142335667
27 BGU GETDE^VA
PATIENT^.0001,.001,.01,.02,.03,.09,.033,F9000001:.001~"41"~DUZ(2
)~"0":4101!.02
28 BGU GENLIST^VA
PATIENT^^1^SC:PAT|BPCSC1^50^SMALL,AMY^SMALL,AMY~^.0001,.001,.0
1,.02,.03,.09,.033,F9000001:.001~"41"~DUZ(2)~"0":4101!.02^^^1142342277^
29 BPC CHARTDATA^25241^2^10^10/1/2000^10/1/2001
30 BGU GETDE^V MEDICATION^.01,.05,.06,.07,1101,1102,1202,.03
31 BGU GENLIST^V
MEDICATION^^1^SC:PCCMED|BPCSC1^25^25241^10/1/2000`10/1/2001`10`
0`2^.01,.05,.06,.07,1101,1102,1202,.03^^^1142346533^
32 BGU DESTRUCTOR^1142346533
33 BGU DESTRUCTOR^1142346533
34 BGU GETDE^V MEDICATION^.01,.05,.06,.07,1101,1102,1202,.03
35 BGU GENLIST^V
MEDICATION^^1^SC:PCCMED|BPCSC1^25^25241^10/1/2000`10/1/2001`10`
0`2^.01,.05,.06,.07,1101,1102,1202,.03^^^1142346493^
36 BGU DESTRUCTOR^1142346493
37 BGU DESTRUCTOR^1142346493
38 BGU GETDE^OUTPATIENT SITE^.17
39 BGU GENLIST^OUTPATIENT SITE^^1^C^25^2248^2248^.17^^^1142347414^
40 BGU DESTRUCTOR^1142347414
41 BGU GETDE^APSP CONTROL^2,15
42 BGU GENLIST^APSP CONTROL^^1^B^25^1^1^2,15^^^1142347414^
43 BGU DESTRUCTOR^1142347414
44 BGU GETDE^RX PATIENT STATUS^.01,3,4,5
45 BGU GENLIST^RX PATIENT STATUS^^1^B^9999^^^.01,3,4,5^^^1142347414^
46 BGU DESTRUCTOR^1142347414
47 BGU DESTRUCTOR^1142347414
48 BPC GETPROBLEMS^25241^10/1/2000^10/1/2001
49 BPC GETFACEDATA^25241^
50 BPC GET SD APPT DATA^25241^10/1/2000^10/1/2001
51 BGU GETDE^V RADIOLOGY^.01,.05,.06,1101,.03,1202
52 BGU GENLIST^V
RADIOLOGY^^1^SC:RAD|BPCSC1^25^25241^10/1/2000`10/1/2001`10`0`2^
.01,.05,.06,1101,.03,1202^^^1142348956^
53 BGU DESTRUCTOR^1142348956
54 BGU DESTRUCTOR^1142348956
55 BGU GETDE^V PATIENT ED^.01,.08,.06,.07,.11,.03,1202
56 BGU GENLIST^V PATIENT
ED^^1^SC:EDP|BPCSC1^25^25241^10/1/2000`10/1/2001`10`0`2
^.01,.08,.06,.07,.11,.03,1202^^^1142349367^
57 BGU DESTRUCTOR^1142349367
58 BGU DESTRUCTOR^1142349367
59 BGU GETDE^HEALTH SUMMARY TYPE^.01
60 BGU GENLIST^HEALTH SUMMARY TYPE^^1^B^25^ADULT REGULAR^ADULT
REGULAR~^.01^^^11
42349777^
61 BGU DESTRUCTOR^1142349777
62 BGU DESTRUCTOR^1142349777
63 BGU GETDE^BI SITE PARAMETER^.01,.11,.15,.09
64 BGU GENLIST^BI SITE PARAMETER^2248^^^^^^.01,.11,.15,.09^^^1142350528^
65 BGU DESTRUCTOR^1142350528
66 BGU DESTRUCTOR^1142350528
67 BPC GET IMM HISTORY DATA^25241
68 BPC GET IMM FORECAST DATA^25241
69 BGU DESTRUCTOR^1142342277
```

```
START FROM:

SEQUENCE NUMBER: 3991                    APPLICATION ID: OE-2
  $J: 24                                 DATE/TIME: SEP 28, 2001@13:54:28
 BUFFER STRING:   BPC GETFACEDATA^25241^
 XQOR STRING:   GETFACE^BPCRC8("25241")
 RETURN DATA:   DATA`SMALL,AMY`2950607`2991201`537973815`Verified by
 SSA`FEMALE`2950110`6`YRS`INDIAN/ALASKA NATIVE`CROW AGENCY``245-2116```CROW
 AGENCY`MONTANA`59022`638-7218`639-2333`CROW TRIBE OF MONTANA`UNSPECIFIED...
 PI`1`MAILHANDLERS BENEFIT PLAN^516642549^2950607^2961012 REGS`2`TALL GRASS
 HEALTH CENTER^33786~SHORT GRASS HEALTH STATION^33786 ADD`1`SS CARD ON FILE
 9-11-95 (ANP)


SEQUENCE NUMBER: 4009                    APPLICATION ID: OE-2
  $J: 24                                 DATE/TIME: SEP 28, 2001@13:54:30
 BUFFER STRING:   BPC GET IMM HISTORY DATA^25241
 XQOR STRING:   GETIHDSP^BPCIMDSP("25241")
 RETURN DATA:   25 21-Jul-1995 @14:05^1-DTP^^^Demo Hospital^^^DIPHTHERIA,
 TETANUS TOXOIDS & PERTUSSIS^103^ 26-Sep-1995 @08:20^2-DTP^^^Demo
 Hospital^^^DIPHTHERIA, TETANUS TOXOIDS & PERTUSSIS^103^ 21-Feb-1996
 @15:00^3-DTP^^^Demo Hospital^^^DIPHTHERIA, TETANUS TOXOIDS & PERTUSSIS^103^
 31-Dec-1996 @10:55^4-DTP^^^Demo Hospital^^^DIPHTHERIA, TETANUS TOXOIDS &
 PERTUSSIS^103^ 30-Aug-2000 @17:11^5-DTaP^^^Demo Hospital^^^DIPHTHERIA,
TETANUS
 TOXOIDS & aPERTUSSIS^133^ 08-Jun-2001 @1:00^6-DT-PEDS^Vomiting^^Demo
 Hospital^^^DIPHTHERIA & TETANUS TOXOIDS, PEDIATRIC^123^1859 21-Jul-1995
 @14:05^1-OPV^^^Demo Hospital^^^POLIOVIRUS VACCINE, LIVE, ORAL^106^ 26-Sep-
1995
 @08:20^2-OPV^^^Demo Hospital^^^POLIOVIRUS VACCINE, LIVE, ORAL^106^ 31-Dec-
1996
 @10:55^3-OPV^^^Demo Hospital^^^POLIOVIRUS VACCINE, LIVE, ORAL^106^ 30-Aug-
2000
 @17:11^4-IPV^^^Demo Hospital^^^POLIO VIRUS, INACTIVATED^107^ 21-Jul-1995
 @12:00^1-HIBTITER^^^Demo Hospital^^^H INFLUENZAE B, HBOC CONJ^126^ 26-Sep-
1995
 @12:00^2-HIBTITER^^^Demo Hospital^^^H INFLUENZAE B, HBOC CONJ^126^ 21-Feb-
1996
 @15:00^3-HIBTITER^^^Demo Hospital^^^H INFLUENZAE B, HBOC CONJ^126^ 31-Dec-
1996
 @10:55^4-HIBTITER^^^Demo Hospital^^^H INFLUENZAE B, HBOC CONJ^126^ 08-Jun-
2001
 @1:00^1-HBIG^^^Demo Hospital^^^HEPATITIS B IMMUNE GLOBULIN^129^ 07-Jun-1995
 @19:30^1-HEP B^^^Demo Hospital^^^HEPATITIS B VACCINE, NOS^110^ 21-Jul-1995
 @14:05^2-HEP B^^^Demo Hospital^^^HEPATITIS B VACCINE, NOS^110^ 21-Feb-1996
 @15:00^3-HEP B^^^Demo Hospital^^^HEPATITIS B VACCINE, NOS^110^ 27-Sep-1996
 @14:10^1-MMR^^^Demo Hospital^^^MEASLES, MUMPS & RUBELLA VIRUS VACCINE^117^
 30-Aug-2000 @17:11^2-MMR^^^Demo Hospital^^^MEASLES, MUMPS & RUBELLA VIRUS
 VACCINE^117^ 27-Sep-1996 @12:00^VARICELLA^^^Demo Hospital^^^VARICELLA VIRUS
 VACCINE^132^ 15-Sep-1998 @12:00^1-HEP A^^^Demo School^^^HEPATITIS A VACCINE,
 NOS^131^ 08-Apr-1999 @12:00^2-HEP A^^^Crow Agency^^^HEPATITIS A VACCINE,
 NOS^131^ 08-Jun-2001 @1:00^3-HEP A^^^Demo Hospital^^^HEPATITIS A VACCINE,
 NOS^131^1859
```

18.0 SAC Requirements/ Exemptions

The following exemptions to the SAC standards have been made for this version.

18.1 TCP/IP Socket Technology Support

Program + Line Number:

```
+48^BGUTCP :   C 56
+34^BGUTCPH :   C 56
+64^BGUTCPL :   .I BGUOS="MSM" C 56
+65^BGUTCPL :   .I BGUOS="OpenM" C BGUTDEV
+72^BGUTCPL :   I BGUOS="MSM" C 56
+73^BGUTCPL :   I BGUOS="OpenM" C BGUTDEV
```

Application Standard:

2.4.3.1 Close, direct use prohibited
Use of O,U,and C commands for device 56, and J (Job M calls) in the BGU namespace.

Program + Line Number:

```
BGUTCPL+46:  ....I BGUUCI'="",BGUVGRP'="" J
EN^BGUTCPH($P(BGUMSG,"^",2)
,$P(BGUMSG,"^",3))[BGUUCI,BGUVGRP]::5 Q

BGUTCPL+47:  ....J EN^BGUTCPH($P(BGUMSG,"^",2),
$P(BGUMSG,"^",3))::5

BGUTCPL+48:  ...J
EN^BGUTCPH($P(BGUMSG,"^",2),$P(BGUMSG,"^",3))::5
```

Application Standard:

2.4.5.1 Job, direct use prohibited
Use of O,U,and C commands for device 56, and J (Job M calls) in the BGU namespace.

Program + Line Number:

```
+44^BGUTCP :   O 56 U 56::"TCP"
+13^BGUTCPH :   O 56 U 56::"TCP"
+13^BGUTCPH :   O 56 U 56::"TCP"
+25^BGUTCPL :   I BGUOS="DSM" O
BGUTSKT:TCPCHAN:5 ;Open listener
```

Application Standard:

2.4.9.1 OPEN, is prohibited
Use of O,U,and C commands for device 56, and J (Job M calls) in the BGU namespace.

Program + Line Number:

```
+44^BGUTCP :   O 56 U 56::"TCP"
+13^BGUTCPH :   O 56 U 56::"TCP"

+63^BGUTCPL :   .I BGUOS="DSM" U BGUTSKT:DISCONNECT ;
release this socket

+71^BGUTCPL :  CLOSE I BGUOS="DSM" U
BGUTSKT:DISCONNECT ; lost connection shut down
```

Application Standard:

2.4.11.1 USE, with parameters with prohibited

Use of O,U,and C commands for device 56, and J (Job M calls) in the BGU namespace.

Reason for Exemption(s): The Kernel Device manager does not support TCP/IP Socket Technology. The MSM operating system uses device 56 as a special device and requires a port number assigned to both the sending and the receiving process. The two port numbers create a unique socket. Because of the kernel limitation direct use of the Open, Use and Close M language commands are required in the BGU namespace. BGUTCPL is the routine which controls the TCPIP listener. With successful silent sign on, the listener will spawn a job called BGUTCPH which is the handler. Use of the O,U,C and J M commands are used in place of the missing kernel device handlers. When the listener spawns the handler, port numbers are negotiated with the Microsoft Winsock TCPIP Interface DLL until unique IP addresses and port numbers are obtained. This set of unique IP addresses and ports creates a TCPIP socket and assures confidentiality for the session. The Kernel Taskman routinely is used for job tasking. Because of the special handshaking requirements with TCPIP Socket management, taskman cannot be used. Where appropriate we have added the code for specific OS conditions.

18.2 Star Reads

Program + Line Number: Consider use of DIR instead of (suspected) READ

+22^BGUTCPH : .R *BGUTLEN1:BGUDTIME I '$T S BGUTBUF="#BYE#" Q

+23^BGUTCPH : .R *BGUTLEN2:BGUDTIME I '$T S BGUTBUF="#BYE#" Q

Applicable Standard: 2.9.1.1 FileMan Utility Routine – DIR
Use of *R (star read) 2. Star or pound READ used

Reason for Exemption: The TCPIP ANSI Standard supports the non visible ASCII character set. Actions within the TCPIP protocol are triggered by individual characters and are not routinely completed with the characters 13 and 10, the carriage return and line feed characters. Further, the traditional "!" (M carriage return, line feed write command) flushes the buffer in TCPIP. Special control characters are used in place of all standard M writes to device 56.

18.3 Security Keys => nsZ

Package name space history: BPC is the namespace assigned to IHS for the Graphical User Interface (GUI) Patient Chart Application using remote procedure calls in and out of RPMS. The equivalent namespace in VA is CPRS. Patient Chart uses BGU routines to handle the initial TCP/IP listener and sign on security. When the external interface successfully signs in to RPMS, session jobs are spawned to create unique partition activity. BPC name spaced remote procedures and routines are used to move raw data between the server and the client. When possible Patient Chart uses the BGU namespace set of core routines which emulate a DAO inside of M. This process minimizes and standardizes code in the client and takes advantage of object technology concepts.

Program + Line Number: Security Key 'BPCBWD', namespace not followed by "Z".
Security Key 'BPCBWP', namespace not followed by "Z".
Security Key 'BPCEDD', namespace not followed by "Z".
Security Key 'BPCEDP', namespace not followed by "Z".
Security Key 'BPCFSP', namespace not followed by "Z".
Security Key 'BPCHSD', namespace not followed by "Z".
Security Key 'BPCHSP', namespace not followed by "Z".
Security Key 'BPCHST', namespace not followed by "Z".

Security Key 'BPCIMA', namespace not followed by "Z".
Security Key 'BPCIMD', namespace not followed by "Z".
Security Key 'BPCIMDF', namespace not followed by "Z".
Security Key 'BPCIMDH', namespace not followed by "Z".
Security Key 'BPCIME', namespace not followed by "Z".
Security Key 'BPCLIL', namespace not followed by "Z".
Security Key 'BPCLILC', namespace not followed by "Z".
Security Key 'BPCLILP', namespace not followed by "Z".
Security Key 'BPCLILR', namespace not followed by "Z".
Security Key 'BPCLM', namespace not followed by "Z".
Security Key 'BPCLO', namespace not followed by "Z".
Security Key 'BPCLOCM', namespace not followed by "Z".
Security Key 'BPCLOLA', namespace not followed by "Z".
Security Key 'BPCLOLC', namespace not followed by "Z".
Security Key 'BPCLOLO', namespace not followed by "Z".
Security Key 'BPCLOOO', namespace not followed by "Z".
Security Key 'BPCLOPO', namespace not followed by "Z".
Security Key 'BPCLOSP', namespace not followed by "Z".
Security Key 'BPCLOWC', namespace not followed by "Z".
Security Key 'BPCLP', namespace not followed by "Z".
Security Key 'BPCLT', namespace not followed by "Z".
Security Key 'BPCLTE', namespace not followed by "Z".
Security Key 'BPCLTP', namespace not followed by "Z".
Security Key 'BPCMNML', namespace not followed by "Z".
Security Key 'BPCMNMLP', namespace not followed by "Z".
Security Key 'BPCMNMLR', namespace not followed by "Z".
Security Key 'BPCMNPLU', namespace not followed by "Z".
Security Key 'BPCMNPO', namespace not followed by "Z".
Security Key 'BPCMNTN', namespace not followed by "Z".
Security Key 'BPCMNUO', namespace not followed by "Z".
Security Key 'BPCMSA', namespace not followed by "Z".
Security Key 'BPCMSE', namespace not followed by "Z".
Security Key 'BPCMSG', namespace not followed by "Z".
Security Key 'BPCMSS', namespace not followed by "Z".
Security Key 'BPCMSTP', namespace not followed by "Z".
Security Key 'BPCNTNH', namespace not followed by "Z".
Security Key 'BPCPATPL', namespace not followed by "Z".
Security Key 'BPCPC', namespace not followed by "Z".
Security Key 'BPCPRA', namespace not followed by "Z".
Security Key 'BPCPRD', namespace not followed by "Z".
Security Key 'BPCPRE', namespace not followed by "Z".
Security Key 'BPCPRL', namespace not followed by "Z".
Security Key 'BPCPRNA', namespace not followed by "Z".
Security Key 'BPCPRNC', namespace not followed by "Z".
Security Key 'BPCPRNR', namespace not followed by "Z".
Security Key 'BPCPRP', namespace not followed by "Z".
Security Key 'BPCPVP', namespace not followed by "Z".
Security Key 'BPCRXC', namespace not followed by "Z".
Security Key 'BPCRXE', namespace not followed by "Z".
Security Key 'BPCRXI', namespace not followed by "Z".
Security Key 'BPCRXMP', namespace not followed by "Z".
Security Key 'BPCRXMPD', namespace not followed by "Z".
Security Key 'BPCRXMPP', namespace not followed by "Z".
Security Key 'BPCRXMPR', namespace not followed by "Z".
Security Key 'BPCRXN', namespace not followed by "Z".

Security Key 'BPCRXPMD', namespace not followed by "Z".
Security Key 'BPCRXPMP', namespace not followed by "Z".
Security Key 'BPCRXPMR', namespace not followed by "Z".
Security Key 'BPCRXR', namespace not followed by "Z".
Security Key 'BPCSDP', namespace not followed by "Z".
Security Key 'BPCSPDA', namespace not followed by "Z".
Security Key 'BPCSPDS', namespace not followed by "Z".
Security Key 'BPCSPP', namespace not followed by "Z".
Security Key 'BPCSPRC', namespace not followed by "Z".
Security Key 'BPCSPTR', namespace not followed by "Z".
Security Key 'BPCSPTRP', namespace not followed by "Z".
Security Key 'BPCTAP', namespace not followed by "Z".
Security Key 'BPCTCV', namespace not followed by "Z".
Security Key 'BPCTEP', namespace not followed by "Z".
Security Key 'BPCTFS', namespace not followed by "Z".
Security Key 'BPCTHS', namespace not followed by "Z".
Security Key 'BPCTIM', namespace not followed by "Z".
Security Key 'BPCTLR', namespace not followed by "Z".
Security Key 'BPCTMS', namespace not followed by "Z".
Security Key 'BPCTPC', namespace not followed by "Z".
Security Key 'BPCTPR', namespace not followed by "Z".
Security Key 'BPCTPV', namespace not followed by "Z".
Security Key 'BPCTRG', namespace not followed by "Z".
Security Key 'BPCTRX', namespace not followed by "Z".
Security Key 'BPCTSP', namespace not followed by "Z".
Security Key 'BPCTTN', namespace not followed by "Z".
Security Key 'BPCTWH', namespace not followed by "Z".
Security Key 'BPCTXR', namespace not followed by "Z".
Security Key 'BPCXRP', namespace not followed by "Z".

Applicable Standard: APNDX A.2.2.1 Security Keys => nsZ

Reason: The "Z" as the forth or fifth character identifies the security key as a lock in association with menu management and the option file. The above keys do not correspond to menu options and locks. Rather, they provide Tag Property values for forms and controls used in the client application. To segregate security key use associated with menus and options from this new GUI related use, the "Z" was not used. Changing GUI Tag Property values and underlining code within the client executable to use the "Z" even when there is no corresponding menu options would cause substantial delays in releasing the application to the field.

19.0 Appendix A: MUMPS Data Access Objects (MDAO)

1.1.1.1.1.1 Introduction

The **MDAO.dll** file is an **in process Active-X Automation Server** used by a Visual Basic program that references it. This automation server exposes two objects: the **CServer** object and the **CMFile** object. The CServer object contains methods and properties which pertain to managing a connection to a MUMPS database server. The CMFile object works in conjunction with the CServer object . It contains methods and properties which allow the VB programmer to easily fetch ,add,change or delete data in a fileman based MUMPS file on the MUMPS server.

Referencing MDAO Objects

To use the MDAO objects, the MDAO.dll needs to be on your machine and registered properly in the registry. Once this is done, the project can't declare and use the objects as above until the MDAO is referenced as follows:

Referencing MDAO in VB4

1. Click on the **References** option within the **Tools** menu.
2. Find the **MUMPS Data Access Objects** item in the **Available References** listbox and check the checkbox next to this reference by clicking on it. This reference must be checked for the project to properly reference MDAO. If this item is not there, more than likely the MDAO.dll has not be registered properly.
3. Click on the **OK** button to complete the referencing process.

Referencing MDAO in VB5

1. Click the **References** option within the **Project** menu.
2. Find the **MUMPS Data Access Objects** item in the **Available References** listbox and check the checkbox next to this reference by clicking on it. This reference must be checked for the project to properly reference MDAO. If this item is not there, more than likely the MDAO.dll has not be registered properly.
3. Click on the OK button to complete the referencing process.

Referencing MDAO in VB6

1. Click the **References** option within the **Project** menu.
2. Find the **MUMPS Data Access Objects** item in the **Available References** listbox and check the checkbox next to this reference by clicking on it. This reference must be checked for the project to properly reference MDAO. If this item is not there, more than likely the MDAO.dll has not be registered properly.
3. Click on the OK button to complete the referencing process.

Declaring MDAO Objects

Declaring CServer

The CServer object can be declared locally using the **Dim** keyword or globally using the **Public** key word. It can be declared in one of two ways.

Dim objServer as New CServer Or Dim objServer as CServer
When the **New** keyword is used as in the first example, an instance of the object is created immediately. If the New keyword is not used in the declaration, an instance of the object must be created before it is used as follows**: Set objServer = New CServer**

Declaring CMFile

Theoretically , the CMFile object can be declared as explained above, but in reality **IT MUST NEVER BE DECLARED USING THE NEW KEYWORD** because if it is,it will never be connected properly to the CServer object and thus will be useless. An example of a correct locally declared CMFile object is:

Dim objFile as CMFile
The above declaration does not instantiate the objFile object, however. **IT MUST BE INSTANTIATED AS SHOWN IN THE FOLLOWING EXAMPLE** for the file object to function correctly:

Set objFile = objServer.CreateFileObject
The CreateFileObject method of the CServer object both instantiates the CMFile object and connects it to the MUMPS database server through the CServer object.

The examples contained in this document assume the above declarations and instantiations have been made and will use the name **objServer** to represent the CServer object and **objFile** for the CMFile object.

The CServer Object

CServer Properties

AccessCode As String (Write Only)

This property is the access code value to connect to the server. If it or the VerifyCode property is a null string, an access/verify dialog box is popped up by the object

Example: `objServer.AccessCode = txtAccessCode.text`

VerifyCode As String (Write Only)

This property is the verify code value to connect to the server. If it or the AccessCode property is a null string, an access/verify dialog box is popped up by the object.

Example: `objServer.VerifyCode = txtVerifyCode.text`

IsOpen As Boolean (Read Only)

This property is used to determine if the connection with the server is open or not.

Example: `If objServer.IsOpen  then DoSub`

ServerPort As Long

This property holds the port number on which a connection is requested. It normally defaults to 8000 but can be set to another value by the program before a connection request is made.

Example: `lblServerPort.Caption = Str$(objServer.ServerPort)`

Server As String

This property is the Name or the IP address of the server to which the connection will be made. If it is a null string, a server address dialog box is popped up by the object.

Example: `objServer.Server = txtAddress.text`

UserName As String (Read Only)

This property contains the user name of the person logged onto the server. This property is set by the object when connection to the server is successful.

Example: `lblUser.caption = objServer.UserName`

DUZ As String (Read Only)

This property contains the IEN (Internal Entry Number) of the person logged onto the server. This property is set by the object when connection to the server is successful.

Example: `lblDUZ.caption = objServer.DUZ`

DUZ2 As String (Read Only)

This property contains the IEN (Internal Entry Number) of the site onto which the user is logged.

Example: `lblDUZ2.caption = objServer.DUZ2`

Facility As String (Read Only)

This property contains the facility name for the server onto which you are logged. It is set by the object after a successful connection is made to the server.

Example: `lblLogInFacility.Caption = objServer.Facility`

LocalAddress As String (Read Only)

This property contains the IP address of the current client machine connected to the server. It is set by the object after a successful connection is made.

Example : `lblLocalAddress.Caption = objServer.LocalAddress`

CardAddress (Read Only)

This property contains the address code for the ethernet card installed in the client's machine. The object gets this address from the client's registry when connection to the server is successful.

Example: `lblEthernetAddress.caption = objServer.CardAddress`

ConnectedPort As Long (Read Only)

This property contains the port number which the server assigned to the object for communication with it . It is set after connection to server is successful.

Example: `lblConnectedPort.Caption = objServer.ConnectedPort`

IHSAppID As String

This property contains the Application ID for the client program in use.

Example: `lblApplicationID.caption = objServer.IHSAppID`

UCI As String

This property is used strictly for program testing and debugging purposes. Normally a handler is jobbed in the UCI where the listener resides and this would be the case at a real site. However, there may be times in development when necessary data on the server resides on other UCIs than the listener's and instead of shutting down the listener in one UCI and starting it up again in another or installing extra data and routines in all the UCIs, this property can be set with the UCI and Volume Group for which you want a handler to be jobbed.

Example: `objServer.UCI = "LBP,DEV"`

AutoConnect As Boolean

This property enables or disables autoconnection to the server. If it is set to true and another program on the client is connected to the server, the current program will connect automatically to the server without prompting for access/verify code. If it is set to false, the user will have to enter access/verify codes every time an application is started.

Example: `objServer.AutoConnect = True`

Connections As integer (Read Only)

This property keeps track of the number of current connections to the server . Only those programs that have set the AutoConnect property to true will be counted.

Example: `lblConnections.Caption = Str$(objServer.Connections)`

DeveloperMode as Boolean

This property should only be used in development. When it is set to true, the connection count kept in the registry never goes below one. Therefore, all applications that have AutoConnect set to true will automatically connect to the server without having to enter an access/verify code.

Example: `objServer.DeveloperMode = True`

LastError as String (Read Only)

This property returns the text of the last error. Reading this property automatically clears it. So using this property as follows will not work:

If `objServer.LastError<>"" then Lib.ErrMsg objServer.LastError` ← **Nothing Will Be Displayed**

Instead set a string variable to the property and then display the variable as shown below:

sErr=objServer.LastError ←**Put the error in a string variable**
If sErr<>”” then Lib.ErrMsg sErr ← **Display the variable if necessary**

CServer Methods

Function Connect() as Boolean

This method is used to request a connection to the server. If it returns true the request was successful; otherwise, the connection failed.

Example:

```
bRet = objServer.Connect
If bRet = True Then
   sUser = objServer.UserName
   sAddress = objServer.Server
   panStatus.Caption = sUser & “ In “ & objServer.Facility
   cbServer.SetFocus
Else
   Lib.ErrMsg “Connection Failed!”
End If
```

Function Disconnect() as Boolean

This method is used to close the connection to a server. It returns true if the close process was successful; otherwise, the close process failed.

Example: `If objServer.IsOpen Then bRet = objServer.Disconnect`

Function Execute(sString as String) as String

The Execute method sends an RPC string synchronously to the server. An RPC string consists of an RPC name field followed by the parameters to the call delimited by “^”; for example, **“BGU GETDIAGNOSISLIST^DIAB^50^^”.** The return value is a character 30 delimited string. The first piece is the number of records following in the string. If this piece is empty or “-1,” then an error has occurred and is in the second piece.

Example:

```
sSendString = “BPC GETDIAGNOSISLIST^DIAB^50^^”
Screen.MousePointer = CR_HOURGLASS
sResult = objServer.Execute(sSendString)
Screen.MousePointer = CR_DEFAULT

If sResult = “” Then
    txtResult.Text = “No Data Returned!”
Exit Sub
End If
If Lib.GetPiece(sResult,Chr$(30),1,1)=”-1” Then
  TxtResult.Text =  Lib.GetPiece(sResult,Chr$(30),2,2)
End If
```

Sub ClearConnectionCount()

This method clears the connection data kept in the registry and sets the connection count to zero. It can only be called when the object is disconnected.

Example:

```
Private Sub cmdClear_Click()
      objServer.Disconnect
      objServer.ClearConnectionCount
      cmdSet_Click
End Sub
```

Function CreateFileObject() as CMFILE

This method is used to create a file object. As explained above **,this is the only way that a useful file object can be created**. The file object created with this method is automatically connected to the server through the server object that created it.

Example:

```
Dim objFile as CMFile
Set objFile = objServer.CreateFileObject
```

Function EncriptString(sString As String, nParam As taskEncript) As String

This method is used to encript a string entered as an access/verify code or electronic signature. The nParam parameter can be one of two possible public enumerated constant values:

taskVerify - indicates that sString should be encripted using the Access/Verify alogorithm.
taskSig – indicates that sString should be encripted using the Electronic Signature alogorithm.

Example:

```
Private Function CkElectSig() As Boolean
  Dim bRet As Boolean
  Dim sSig As String

  bRet = False
  sSig = objS.EncriptString(txtSig.Text, taskSig)
  If sSig = sElectSig Then bRet = True
  CkElectSig = bRet
End Function
```

The CMFile Object

CMFile Properties

File as String

This property represents the Fileman file name or number. It must be set before the OpenFile method is called.

Example: `objFile.File = “2”  or  objFile.File = “VA PATIENT”`

IEN as Long

If you wish to obtain data from a specific record in the file, the IEN property should be set to IEN (Internal Entry Number) of the record you wish . If used, it must be set before the OpenFile method is called.

```
     Example:

objFile.File = “VA PATIENT”

 OBJFILE.IEN = 1
     objFile.Field(“Patient Name”) = “.01”
     bRet = objFile.OpenFile
```

Xref as String

Use this property to tell the object what Fileman cross reference to use to get your data. This property is ignored if the IEN property is set. If this property is not set , data is obtained from the file starting at the first IEN. If used, it must be set before the OpenFile method is called.

Example:

```
objFile.File = “VA PATIENT”
objFile.Xref = “B”
objFile.Field(“Patient Name”) = “.01”
bRet = objFile.OpenFile
```

Max as Integer

This property is the maximum number of records to return at a time. The default maximum is 25. This value is ignored if the IEN property is set. If used, it must be set before the OpenFile Method is called.

Example:

```
objFile.File = “VA PATIENT”
objFile.Max = 50
objFile.Field(“Patient Name”) = “.01”
bRet = objFile.OpenFile
```

Start as String

This property is used in conjunction with the Finish and Xref properties to indicate a range of records to obtain from the file. For example, you may wish to pick up a list of patient records whose last names start with “SM”. Say the VA PATIENT file’s “B” cross reference is used for patient names. The following code could then accomplish the above task:

```
objFile.File = “VA PATIENT”
objFile.Xref = “B”
objFile.Start =  “SM”
objFile.Finish =  “SM~”
objFile.Field(“Patient Name”) = “.01”
bRet = objFile.OpenFile
```

If used, it must be set before the OpenFile method is called.

Finish as String

See above.

Direction as String

Does Nothing Currently

Screen as String

This property contains MUMPS code that screens the data on the server side. If used, it must be set before the OpenFile method is called

Example:

```
objFile.File = "V POV"
objFile.XRef = "AC"
objFile.Start = “1”
objFile.Finish = “1”
objFile.Screen = "BGUV(9000010.07,.03)>2921231.2359"
objFile.Field("DX Code") = ".01"
objFile.Field(“Visit”) = “.03”

bRet = objFile.OpenFile
```

The above code picks up visit data occuring after 12/31/92 at 23:59 on patient #1.

Field(sFieldName as string) as String

This property is used to assign a name or identifier to a field in a Fileman file on the server side. It also tells the object to get the field’s definition and value(s) from the server. The field’s value can then be retrieved or set later using the **Value** property. The fieldname assigned is not case sensitive. **At least one field must be assigned before the OpenFile method can be called sucessfully.** There are four ways (representing different Fileman types of data) that the Field property can be set:

Non-Multiple Fields:

```
objFile.Field(“Patient Name”) = “.01”
```

In this example, .the 01 field value is collected from the server file. This represents one value for each record obtained. If the field is a pointer, the value returned is the resolved value not the pointer value itself unless the field is used later as an input field in a lookup or a pointer value is explicitly requested as follows:
objFile.Field(“Patient IEN”) = “.01-P”

Multiple or Word Processing Fields:

```
ObjFile.Field(“Multiple Data”) = “.03!.01”
```

In this example, the .01 field value is collected from the .03 multiple field. This can represent one to many values for each record obtained. Any level of multiples within multiples can be represented in this way. For Example, multiple data four levels deep would be represented in this way: `objFile.Field(“Mult In Mult”) = “.03!.01!.02!.01”.` Pointer values are resolved as explained above.

Fields Obtained From A Lookup Into Another File:

Non-multiple fields as input and output

```
objFile.Field("Lookup") = "F80:.01~""0"":3"
```

In this example, the .01 field from the current file (the input) is used as a lookup into File 80 at the “0” node level and returns field 3 (the output) from that file. This returns one value per record..

Multiple field as input and non-multiple field as output

```
objFile.Field("Issue Date") = F52:52!.01~""0"":1"
```

This field definition uses a multiple value (52!.01) as the input into file 52 and returns the non-multiple field 1. Many values could be returned for this field.

Multiple fields as input and output

```
objfile.Field("Refill Date") = "F52:52!.01~""1"":52!.01"
```

This example uses a multiple field as input and returns a multiple field as output. Many values could be returned for this also.

A lookup within a lookup

```
objFile.Field("Drug") = "F50:52;6~""0"":.01"
```

This example gets the .01 field from File 50 using the 6 field from File 52 as input. However, in order for this to work, the lookup for 52;6 must be set up first as follows:

```
ObjFile.Field(“Drug Pointer”) = “F52:52!.01~””0””:6”
```

Pointer fields returned in a lookup are not resolved. The number in the double quotes preceeded by the “~” in each of the above examples (i.e ~””0””) is required for all lookup type field assignments. This number is the last literal node at the level from which data is obtained. The double set of quotes are required for the lookup to work correctly. **Lookup fields are read only. Their values cannot be set.**

Computed Or Special Fields:

A computed field definition can be represented by any of the above examples. The main difference is that instead of returning data stored on the server, data is returned after it is has been computed on the server .

Special fields are fields defined specifically within the CMFile object. Currently two special fields are available: the **.0001** and the **IEN** field**.** The .0001 field represents the subscript value for each record in the MUMPS cross reference used to collect the data, and the IEN field represents the internal entry number for each record . **The .0001 field must be assigned to the Field property to be used**, while **the IEN field is always available for each record without having to be assigned. Computed and Special Fields are read only. Their values cannot be set.**

Example:

```
      objFile.File = “VA PATIENT”
      objFile.Xref = "B"  ← To use the .0001 field, a cross reference
must be set.
      objFile.Start = "SMITH"
      objFile.Finish = "SMITH~"
      objFile.Field("Subscript") = ".0001"    ← Must explicitly assign
      objFile.Field("Name") = ".01"
      bRet=objFile.OpenFile
      if bRet Then
       objFile.MoveFirst
      sSub = objFile.Value("Subscript")
      sIEN = objFile.Value("IEN")   ←Not assigned but can still
reference it
            sName = objFile.Value("Name")
            objFile.CloseFile
      end if
```

If the first cross reference traversed looks like this: `^DPT("B","SMITH,ABE",304)=""`, then the value of the first record's "SubScript" field would be "SMITH,ABE" and the value in it's "IEN" field would be "304".

Note: Do not assign a field the name "IEN" or there will be problems.

IsFileOpen as Boolean (Read Only)

This property determines whether the file has been opened. This can help the programmer determine which methods or properties can or can't be used.

Example:

```
If objFile.IsFileOpen Then
  If objFile.SeekValue("IEN",sPatIEN) Then
      objFile.Delete

         OBJFILE.UPDATE
  End If
End if
```

RecordCount as integer (Read Only)

This property contains the number of records currently contained within the record buffer. It is not meaningful unless used after the OpenFile method has been called.

Example:

```
If objFile.RecordCount > 0 Then
    objFile.MoveFirst
    Do While Not objFile.EOF
        Set li = lvData.Listitems.Add(,,objFile.Value("Name"))
         li.Subitems(1) = objFile.Value("SSN")
         objFile.MoveNext
    Loop
End If
```

CurrentIndex as integer (Read Only)

This property contains the current record number pointed to in the record buffer by the file object. It is not meaningful unless used after the OpenFile method has been called.

Example:

```
If SeekValue("IEN",sIEN)then nCurrentRec = objFile.CurrentIndex
```

BookMark as integer

This property returns or sets a value that identifies a record in the objects record buffer. Setting a bookmark sets the internal record pointer to it's value. If it is set to a value that doesn't make sense (<1 or > number of records) the internal pointer does not change.

Example:

```
nOldBookMark = objFile.BookMark
If SeekValue("IEN",sIEN) then objFile.BookMark = objFile.CurrentIndex
      .
      .
      objFile.BookMark = nOldBookMark
```

IsMore As Boolean (Read Only)

The value of this property is used to determine whether there is more data to be retrieved from the Server. It is usually used in conjunction with the GetMore method.

Example: `If objFile.IsMore Then bRet = objFile.GetMore`

BOF as Boolean (Read Only)

This property indicates whether the current record position within the record buffer is before the first record in the buffer. You can use the BOF and EOF properties to determine whether the file object contains records or whether you've gone beyond the limits of the record buffer when you move from record to record. If the record buffer contains no records, BOF and EOF are both True.

Example:

```
objFile.MoveLast
      While not objFile.BOF
            objFile.MovePrevious
      Wend
```

EOF as Boolean (Read Only)

This property indicates whether the current record position within the record buffer is after the last record in the buffer. You can use the BOF and EOF properties to determine whether the file object contains records or whether you've gone beyond the limits of the record buffer when you move from record to record. If the record buffer contains no records, BOF and EOF are both True.

Example:

```
objFile.MoveFirst
      While not objFile.EOF
```

```
                                        objFile.MoveNext
                                  Wend
```

Value(sFieldName as String,sIENS as String) as String

This property is used to retrieve or alter field data in the CMFile object. Both parameters to the call are required within the VB4.0 environment, but the sIENs parameter is an optional parameter within the VB5 environment. The sIENS parameter is only meaningful when retrieving or setting data from Fileman multiple fields or retrieving data from lookup fields pointed to by a multiple field. The sIENs parameter is ignored with other field types (however, within VB4.0 it must always be set to some value). **Note: Field Values from lookup fields and special fields are read only. An error occurs if you try to set them.**

See "Setting Multiple Values" section for a detailed information about setting those values.

VB4.0 Examples:

```
sCulture = objFile.Value("CULTURE", "0")   ← The "0 is an arbitrary value.
sOrganism = objFile.Value("ORGANISM", "0")

objVFile.AddNew
objVFile.Value("Drug", "0") = objFile.Value("Drug", "0")
objVFile.Update

sSub = sSelIEN & ",+1"   ←Setting a new value into a multiple
objFile.Edit
objFile.Value("Refill Date", sSub) = sNow
objFile.Update
```

VB5 Examples:

```
sCulture = objFile.Value("CULTURE")      ← 2nd parameter isn't needed
sOrganism = objFile.Value("ORGANISM",)

objVFile.AddNew
objVFile.Value("Drug") = objFile.Value("Drug")
objVFile.Update

sSub = sSelIEN & ",+1"  ←Setting a new value into a multiple
objFile.Edit
objFile.Value("Refill Date", sSub) = sNow ← 2nd parameter needed
objFile.Update
```

Keys(sFieldName as String) as String   (Read Only)

This property returns the key values delimited by the "^" character for the field specified by sFieldName. This property is only applicable to Fileman multiple or word processing fields. A null string will always be returned with other types . A key is a comma delimited string consisting of: **FieldName,Record# ,IENS.** The IENS part of the key can be from 2 to many comma delimited values which represent the internal entry numbers for each level of the multiple or word processing field represented. The ValueCount and Keys properties can be used together to access and/or edit the values of multiple or word processing fields.

Example:

```
sKeys = objFile.Keys(“RX Pointer”)
nVals = objFile.ValueCount(“RX Pointer”)
If nVals>0 then Redim sRXIEN(1 to nVals)
For I=nVals to 1 Step -1
 sS = Lib.GetPiece(sKeys, “^”, i, i)
 sSub = Lib.GetPiece(sS, ",", 3, 99) ← Get the IENS part from the key
 sRXIEN(i) = objFile.Value(“RX Pointer”,sSub)  ← Get the value
Next
```

ValueCount(sFieldName as String) as Integer (Read Only)

The property returns the number of values stored for the sFieldName field. This property is only applicable to Fileman multiple or word processing fields and lookup fields whose input fields are multiples. The ValueCount and Keys properties can be used together to access and/or edit multiple or wordprocessing fields (See Above Example). The ValueCount count can also be used as in the following example to access multiple values in a lookup field.

Example:

```
ObjFile.Field(“Drug Lookup”)=”F52:52!.01~””0””:3” ← a multiple input
.
.
nVals = objFile.ValueCount(“Drug Lookup”)
If nVals>0 Then Redim sDrugs(1 to nVals)
For I=1 to nVals
 sSub=”” & I  ← Turns the index into a string
 sDrugs(I)=objFile.Value(“Drug Lookup”,sSub)  ← Get value with index
Next
```

IENS(sFieldName As String) As String (Read Only)

This property returns IENS values delimited by the “^” character for the field specified by sFieldName. This property is only applicable to Fileman multiple or word processing fields. A null string will always be returned with other types . An IENS is comma delimited string which represents the internal entry numbers for each level of the multiple or word processing field represented. The ValueCount and IENS properties can be used together to access and/or edit the values of multiple or word processing fields.

Example:

```
sIENS = objFile.IENS(“RX Pointer”)
nVals = objFile.ValueCount(“RX Pointer”)
If nVals>0 then Redim sRXIEN(1 to nVals)
For I=nVals to 1 Step -1
```

```
 sSub = Lib.GetPiece(sIENS, "^", i, i)
 sRXIEN(i) = objFile.Value("RX Pointer",sSub) ← Get the value
Next
```

FieldSetIENs(nIndex As Integer) As String (Read Only)

This property is used in conjunction with the **CreateFieldSet** method to obtain an IENS value from a field set. Creating a fieldset is fast way to obtain information from a multiple or wordprocessing field. See the **CreateFieldSet** method description later in this document for a detailed explanation of fieldsets.

Example:

```
objFile.File = "NEW PERSON"
 objFile.IEN = Val(objServer.DUZ)
  objFile.Field("Keys") = "51!.01"
  If objFile.OpenFile Then
   If objFile.RecordCount > 0 Then
    n = objFile.CreateFieldSet("Keys")  ←Create a fieldset
          If n > 0 Then ReDim sKeyIENs(1 To n)
    For i = 1 To n
     DoEvents
     sKeyIENs(i) = objFile.FieldSetIENs(i)  ←Get the IENS values
    Next
    End If
   objFile.CloseFile
  End If
```

FieldSetVal(nIndex As Integer) As String (Read Only)

This property is used in conjunction with the **CreateFieldSet** method to obtain a value from a field set. Creating a fieldset is fast way to obtain information from a multiple or wordprocessing field. See the **CreateFieldSet** method description later in this document for a detailed explanation of fieldsets.

Example:

```
  objFile.File = "NEW PERSON"
 objFile.IEN = Val(objServer.DUZ)
  objFile.Field("Keys") = "51!.01"
  If objFile.OpenFile Then
   If objFile.RecordCount > 0 Then
    n = objFile.CreateFieldSet("Keys")  ←Create a field set
          If n > 0 Then ReDim sKeys(1 To n)
    For i = 1 To n
     DoEvents
     sKeys(i) = objFile.FieldSetVal(i)  ←Get the values
         Next
   End If
   objFile.CloseFile
  End If
```

LastError as String (Read Only)

This property returns the text of the last error. Reading this property automatically clears it. So using this property as follows will not work:

`If objFile.LastError<>"" then Lib.ErrMsg objFile.LastError` ← **Nothing Will Be Displayed**

Instead set a string variable to the property and then display the variable as shown below:

`sErr=objFile.LastError` ←**Put the error in a string variable**

`If sErr<>"" then Lib.ErrMsg sErr` ← **Display the variable if necessary**

CMFile Methods

Function OpenFile() as Boolean

This method gets field definitions and values for the file object that invokes it. If the call is successful, True is returned, otherwise False. **The File property and at least one Field property must be set before this method can be called successfully** .

Example:

```
objFile.File = "VA PATIENT"
objFile.Field("Name") = ".01"
If objFile.OpenFile Then
  .
  .
  .
  objFile.CloseFile
Else
  Lib.ErrMsg "Unable To Open File!"
End If
```

Function CloseFile() as Boolean

This method clears field definitions and values for the file object that invokes it. If the call is successful, True is returned, otherwise False.

Example:

```
objFile.File = "VA PATIENT"
objFile.Field("Name") = ".01"
If objFile.OpenFile Then
  .
  .
  .
  objFile.CloseFile
Else
  Lib.ErrMsg "Unable To Open File!"
End If
```

Function GetMore() as Boolean

This method gets more field values, if there are more to get. .If the call is successful, True is returned, otherwise False. This method should be used in conjunction with the IsMore property as shown in the following example.

Example: `If objFile.IsMore Then bResult = objFile.GetMore`

Function SeekValue(sFieldName as String,sValue as String) as Boolean

This method finds the first record in the record buffer in which sFieldName equals sValue. The search begins at the record following the current one in the record buffer. If a record is found , the current index of the record buffer is set to point at this record, otherwise the current index does not change.

Example:

```
If objFile.SeekValue (“Name”,”DOE,JOHN”)  Then
  Lib.InfoMsg “Record Found!”
End If
```

Sub MoveFirst()

If there are records in the record buffer, this method moves the current index to the first record .

Example : `objFile.MoveFirst`

Sub MoveLast()

If there are records in the record buffer, this method moves the current index to the last record.

Example: `objFile.MoveLast`

Sub MoveNext()

This method moves the current index to the next record in the record buffer. If there are no more records, the EOF property is set to true. MoveNext is used to traverse the record buffer in a forward direction.

Example:

```
objFile.MoveFirst
While Not objFile.EOF
  .
  .
  objFile.MoveNext
Wend
```

Sub MovePrevious()

This method moves the current index to the previous record in the record buffer. If there are no previous records, the BOF property is set to true. MovePrevious is used to traverse the record buffer in a backwards direction.

Example:

```
objFile.MoveLast
While Not objFile.BOF
  .
  .
  objFile.MovePrevious
Wend
```

Sub AddNew()

This method is used in conjunction with the Update method to add a new record to the file. The record is not added until the Update method is performed. The Internal Entry Number for this new record is assigned by FileMan on the server side and it and the other data is stored by the CMFile object after the call is completed by the Update method.

Example:

```
objFile.AddNew ← Tell the object you want to add new data to the file
objFile.Value(“Name”,”0”)=”DOE,JOHN”  ← Set data
objFile.Value(“Sex”,”0”)=”M”
objFile.Value(“SSN”,”0”)=”724567657”
objFile.Update  ← Do the AddNew operation
```

Sub CreateNew(sCreateIEN as String)

This method is similar to the AddNew method except that it allows you to tell Fileman what Internal Entry Number you wish to assign the record (sCreateIEN). This record is not created until the Update method is performed.

Example:

```
sCreateIEN = objPatFile.Value(“IEN”,”0”)
objFile.CreateNew sCreateIEN  ← You want a new record with this IEN
objFile.Value(“Patient Pointer”)=sCreateIen  ← Set the field data

 OBJFILE.VALUE(“RX POINTER”)=SRXIEN
     objFile.Update  ← Do the CreateNew operation
```

Sub Edit()

This method is used in conjunction with the Update method to edit field data for an existing record in the record buffer. New multiple or word processing data can also be added with this method. The actual edit operation is not done until the Update method is performed.

Example:

```
sSub = sSelIEN & ",+1"  ←Setting a new multiple value into this record
objFile.Edit  ← Tell the object you want to edit a record
objFile.Value("Refill Date", sSub) = sNow  ← Set the field Data
objFile.Update ← Do the Edit operation
```

Sub Delete()

This method is used in conjunction with the Update method to delete a record from the file. The actual delete is not done until the Update method is performed.

Example:

```
If objFile.SeekValue(“IEN”,sIENToDelete) Then
  objFile.Delete
  objFile.Update
End If
```

Sub DeleteNode(sFieldName as String,sIENs as String)

This method is used in conjunction with the update method to delete a node or nodes from a multiple or word-processing field. sFieldName is the name of the multiple or word-processing field. If sFieldName is not a multiple or word-processing field, no action is taken. sIENs is a comma delimited string representing the IENs for the level at which the delete is to take place. If sIENs is null, the LastError property is set and no action is taken.

Example:

```
objF.File = "ROLODEX"
    objF.IEN = nIEN
    objF.Field("Pet Name") = "2!.01"

    If objF.OpenFile Then
      nIdx = lstPets.ListIndex

      If sPetIENS(nIdx) <> "" Then
        objF.DeleteNode "Pet Name", sPetIENS(nIdx)
        objF.Update
        sErr = objF.LastError
              if sErr = “” Then
              .
        .
End If
.
```

Sub Update()

This method is used in conjunction with the AddNew, CreateNew, Edit, DeleteNode, and Delete methods to signal the object to perform the requested operation. See the above examples.

Sub CancelUpdate()

This method is used to clear any request using the AddNew, CreateNew, Edit,DeleteNode, or Delete methods.

Any subsequent Update method call performed after CancelUpdate is ignored, until another AddNew, Edit, etc. method is performed.

Example:

```
objFile.AddNew
   objFile.Value(“Name”)=sName
   If bContinue Then
  objFile.Update
Else
  objFile.CancelUpdate
End If
```

Function CreateFieldSet(sFieldName As String) As Integer

This method creates a fieldset for the sFieldName field and returns an integer representing the number of values in the created fieldset. If a zero is returned, no values were in the field or the field was not a multiple or wordprocessing field. Fieldsets can only be created for multiple and wordprocessing fields. A fieldset represents a **“set”** of data pertaining to sFieldName. Currently this set of data consists of the IENS values and the actual values for sFieldName. The properties **FieldSetIENs** and **FieldSetVal** are used to get the data in a fieldset. Fieldsets were implemented in MDAO to address the inefficiencies found when using the ValueCount ,IENS and Keys properties. Therefore**, using fieldsets is the recommended way to obtain multiple/wordprocessing field data.**

Example:

```
  objFile.File = "NEW PERSON"
objFile.IEN = Val(objServer.DUZ)
  objFile.Field("Keys") = "51!.01"
  If objFile.OpenFile Then
    If objFile.RecordCount > 0 Then
      n = objFile.CreateFieldSet("Keys") ←Create a fieldset
    If n > 0 Then ReDim sKeys(1 To n)
        For i = 1 To n
        DoEvents
        sKeys(i) = objFile.FieldSetVal(i)   ←Get the values
      Next
    End If
    objFile.CloseFile
  End If
```

Setting Multiple Values

MDAO uses an IENS string convention similar to the Fileman IENS string convention for UPDATE^DIE except that the order of the comma delimited IENs is reversed and there is no trailing comma. For example a Fileman IENS string to add data two levels deep would be “+2,+1,”, while an MDAO representation of the same string would be “+1,+2”.

Setting New Data For A New Record

To set multiple data for a new record, a placeholder represented by an integer prefixed with a “+” must be used for each level of the record to be set . Each placeholder must be separated by a comma. For example, “+1,+2”. This comma delimited string represents the IENS string as explained above, and it should be the sIENS parameter to the Value property when that property is set. The number after the “+” must be in ascending order left to right. For example, “+2,+1” would be incorrect. The placeholders don’t have to proceed in sequential order, however. So “+2,+11” would be correct. Once a placeholder is defined , it should be used consistently at the same level for the same record. For example, if two multiple values are to be set at the same level and the first IENS string is “+1,+2”, the second multiple should be represented as “+1,+3” not “+3,+4” .

Example: Sets two values for the “Pet Name” multiple and two values for the “Pet Immunizations” multiple under each “Pet Name” multiple.

```
If Not objF Is Nothing Then
    objF.File = "ROLODEX"

  objF.Field("Name") = ".01"
  objF.Field("Pet Name") = "2!.01"
  objF.Field(“Pet Immunizations”)=”2!2!.01”

  If objF.OpenFile Then
   objF.AddNew ←Add a record
   objF.Value("Name") = txtName.Text
     n=1
     For i = 1 To 2
       n=n+1
       sIENS = “+1,+” & n →i=1 → sIENS=”+1,+2”
                                                           →i=2 →
   sIENS=”+1,+5”

       objF.Value(“Pet Name”,sIENS)=sPetName(i) ←Set  value
       For j = 1 to 2
         n = n + 1
         sImmIENS=sIENS & “,+” & n  →i=1 j=1→ sImmIENS=”+1,+2,+3”
                                                                →i=1
   j=2 → sImmIENS=”+1,+2,+4”
                                                                →i=2
   j=1 → sImmIENS=”+1,+5,+6”

   →i=2 j=2 → sImmIENS=”+1,+5,+7”

         objF.Value(“Pet Immunizations”,sImmIENS)=”None” ←Set Value
        Next
      Next
      objF.Update
```

.

Setting New Data In An Existing Record

To set multiple data into an existing record, an IENS string must be constructed consisting of the actual IEN at each level **before** the levels to be set and a placeholder represented by an integer prefixed with a “+” **at** the levels to be set . Each IEN and placeholder must be separated by a comma. For example, “234,+1”. The “234” in this example is the actual IEN of the top node for the file to be set. This comma delimited string represents the IENS string as explained above, and it should be the sIENS parameter to the Value property when that property is set. Numbers after the “+” in placeholders must be in ascending order left to right. For example, the IENS string “234,+2,+1” would be incorrect. The placeholders don’t have to proceed in sequential order, however. So “234,+2,+11” would be correct. Once a placeholder is defined , it should be used consistently at the same level for the same record. For example, if two multiple values are to be set at the same level and the first IENS string is “234,+1,+2”, the second multiple should be represented as “234,+1,+3” not “234,+3,+4” .

Example: Sets two values for the “Pet Name” multiple and two values for the “Pet Immunizations” multiple under each “Pet Name” multiple in an existing record.

```
.
.

If Not objF Is Nothing Then
  objF.File = "ROLODEX"
  objF.IEN = nChosenIEN  ←Get specific IEN  (For this example assume
it’s 25)
  objF.Field("Name") = ".01"
  objF.Field("Pet Name") = "2!.01"
  objF.Field(“Pet Immunizations”)=”2!2!.01”

  If objF.OpenFile Then
   objF.Edit  ←Edit this record
   objF.Value("Name") = txtName.Text
     n = 0
     For i = 1 To 2
       n = n + 1
       sIENS = nChosenIEN &“,+” & n    →i=1 → sIENS=”25,+1”

         →i=2 → sIENS=”25,+4”

       objF.Value(“Pet Name”,sIENS)=sPetName(i) ←Set Value
       For j = 1 to 2
         n = n + 1
         sImmIENS=sIENS & “,+” & n  →i=1 j=1→ sImmIENS=”25,+1,+2”
                                                                →i=1
   j=2 → sImmIENS=”25,+1,+3”
                                                                →i=2
   j=1 → sImmIENS=”25,+4,+5”

   →i=2 j=2 → sImmIENS=”25,+4,+6”

            objF.Value(“Pet Immunizations”,sImmIENS)=”None” ←Set Value
        Next
      Next
      objF.Update
  .
  .
```

Editing Data In An Existing Record

To edit multiple data in an existing record, an IENS string must be constructed using the IEN at each level of the multiple to be edited . Each IEN must be separated by a comma. For example, “234,1”. The “234” in this example is the IEN of the top node and the “1” is the IEN of the subnode to be edited. This comma delimited string represents the IENS string as explained above, and it should be the sIENS parameter to the Value property when that property is set

Example: .

```
If Not objF Is Nothing Then
  objF.File = "ROLODEX"
  objF.IEN = nChosenIEN  ←Get specific IEN  (For this example assume
it’s 25)
  objF.Field("Name") = ".01"
  objF.Field("Pet Name") = "2!.01"
  objF.Field(“Pet Immunizations”)=”2!2!.01”

  If objF.OpenFile Then
   objF.Edit  ←Edit this record
   objF.Value("Name") = txtName.Text
   sIENS = nChosenIEN & “,” & nNameIEN ←Assuming nNameIEN is 1 then

      ←sIENS=”25,1”
   objF.Value(“Pet Name”,sIENS)=sPetName(i) ←Set Value
   For j = 1 to 2
     sImmIENS=sIENS & “,” & j → j=1→ sImmIENS=”25,1,1”
                                                       → j=2 →
   sImmIENS=”25,1,2”
     objF.Value(“Pet Immunizations”,sImmIENS)=”None” ←Set Value
   Next
   objF.Update
```

20.0 Appendix B: Detailed Data Dictionary Listing

The following is the detailed data dictionary for the files used with the IHS GUI applications. Theses files are in the 90061-90069 file range and uses the BGU namespace. There are no additional files used for the Patient Chart application (BPC namespace). Additionally, the REMOTE PROCEDURE file as released by the VA (file 8994) and modified by the IHS is used with the BGU namespace. This file is included in section 10.0.

```
CONDENSED DATA DICTIONARY---BGU TRACE FILE (#90061)UCI: DEV,DEV   VERSION: 1.2

STORED IN: ^BGUTRACE(                                     10/01/01    PAGE 1
------------------------------------------------------------------------------
--

                                                  FILE SECURITY
                                   DD SECURITY    : @       DELETE SECURITY: B
                                   READ SECURITY  : B       LAYGO SECURITY : B
                                   WRITE SECURITY : B
CROSS REFERENCED BY:
      SEQUENCE NUMBER(B)   $J(C)   APPLICATION ID(D)


                                 FILE STRUCTURE

FIELD     FIELD
NUMBER    NAME

.01       SEQUENCE NUMBER (RNJ5,0X), [0;1]
.02       APPLICATION ID (RF), [0;2]
.03       $J (RNJ6,0X), [0;3]
.04       DATE/TIME (RD), [0;4]
.05       BUFFER STRING (Multiple-90061.01), [1;0]
          .01  BUFFER STRING (W), [0;1]
.06       XQOR STRING (Multiple-90061.02), [2;0]
          .01  XQOR STRING (W), [0;1]
.07       RETURN DATA (Multiple-90061.03), [3;0]
          .01  RETURN DATA (W), [0;1]



CONDENSED DATA DICTIONARY---BGU SITE PARAMETERS FILE (#90062)UCI: DEV,DEV
VERS
ION: 1.2

STORED IN: ^BGUSP(                                        10/01/01    PAGE 1
------------------------------------------------------------------------------
--

                                                  FILE SECURITY
                                   DD SECURITY    : @       DELETE SECURITY: B
                                   READ SECURITY  : B       LAYGO SECURITY : B
                                   WRITE SECURITY : B
CROSS REFERENCED BY:
      SITE(B)


                                 FILE STRUCTURE
```

```
FIELD     FIELD
NUMBER    NAME

.01       SITE (RP4.3'), [0;1]
.02       TRACE ALL (RS), [0;2]
.03       $J TO TRACE (Multiple-90062.01), [1;0]
          .01  $J TO TRACE (NJ6,0), [0;1]
          1    Number of sessions to retain (NJ3,0), [0;2]
.04       SOCKET (NJ5,0), [2;1]
.05       STATISTICS (S), [0;3]
.06       SERVER TIMEOUT (NJ4,0), [0;6]
.07       DAYS TO KEEP ENTRIES (NJ3,0), [0;7]



CONDENSED DATA DICTIONARY---BGU SIGN-ON STATISTICS FILE (#90062.1)UCI: DEV,DEV
 VERSION: 1.2

STORED IN: ^BGUSEC(                                       10/01/01    PAGE 1
------------------------------------------------------------------------------
--

                                                  FILE SECURITY
                                   DD SECURITY    : @     DELETE SECURITY: B
                                   READ SECURITY  : B     LAYGO SECURITY : B
                                   WRITE SECURITY : B
CROSS REFERENCED BY:
      SIGNOFF TIME(AC)  USER(CUR)


                                 FILE STRUCTURE

FIELD     FIELD
NUMBER    NAME

.001      DATE/TIME (D), [ ]
.01       USER (RP200'), [0;1]
1         DEVICE $I (F), [0;2]
2         JOB (NJ7,0), [0;3]
3         SIGNOFF TIME (D), [0;4]
4         NODE NAME (F), [0;5]
5         SOCKET (F), [0;7]
6         ETHERNET ADDRESS (F), [0;8]
7         APPLICATION ID (F), [0;6]
10        PARAMETER (F), [0;10]
20        CALL STATISTICS (Multiple-90062.11), [20;0]
          .01  CALL STATISTICS (MF), [0;1]
          .02  NUMBER (NJ3,0), [0;2]
97        ELAPSED TIME (SECONDS) (CJ10), [ ; ]
99        ELAPSED TIME (MINUTES) (CJ8X), [ ; ]



CONDENSED DATA DICTIONARY---BGU SERVER FILE (#90063)UCI: DEV,DEV   VERSION:
1.2

STORED IN: ^BGUSV(                                        10/01/01    PAGE 1
------------------------------------------------------------------------------
--
```

```
                                                  FILE SECURITY
                                   DD SECURITY    : @     DELETE SECURITY: B
                                   READ SECURITY  : B     LAYGO SECURITY : B
                                   WRITE SECURITY : B
CROSS REFERENCED BY:
      NAME(B)


                                 FILE STRUCTURE

FIELD     FIELD
NUMBER    NAME

.01       NAME (RF), [0;1]
1         PORT (F), [0;2]




CONDENSED DATA DICTIONARY---BGU TABLE FILE (#90064)UCI: DEV,DEV   VERSION: 1.2

STORED IN: ^BGUT(                                          10/01/01    PAGE 1
-------------------------------------------------------------------------------
--

                                                  FILE SECURITY
                                   DD SECURITY    : @     DELETE SECURITY: B
                                   READ SECURITY  : B     LAYGO SECURITY : B
                                   WRITE SECURITY : B
CROSS REFERENCED BY:
      NAME(B)


                                 FILE STRUCTURE

FIELD     FIELD
NUMBER    NAME

.01       NAME (RF), [0;1]
1         FILE NUMBER (F), [0;2]
2         FIELD (Multiple-90064.02), [1;0]
          .01  FIELD (MF), [0;1]
          1    FIELD NUMBER (F), [0;2]



CONDENSED DATA DICTIONARY---BGU EXECUTE CODE FILE (#90064.1)UCI: DEV,DEV
VERSI
ON: 1.2

STORED IN: ^BGUEXC(                                        10/01/01    PAGE 1
-------------------------------------------------------------------------------
--

                                                  FILE SECURITY
                                   DD SECURITY    : @     DELETE SECURITY: B
                                   READ SECURITY  : B     LAYGO SECURITY : B
                                   WRITE SECURITY : B
CROSS REFERENCED BY:
      NAME(B)  EXECUTE CODE(C)  ACTIVATION(D)


                                 FILE STRUCTURE

FIELD     FIELD
NUMBER    NAME

.01       NAME (RF), [0;1]
1         EXECUTE CODE (FX), [.1;E1,220]
2         ACTIVATION (S), [0;2]
3         TYPE (S), [0;3]
4         ERROR PROMPT (F), [.2;1]
5         APPLICATION NAME (P9.4'), [2;1]
6         DESCRIPTION (Multiple-90064.101), [1;0]
          .01  DESCRIPTION (W), [0;1]



CONDENSED DATA DICTIONARY---BGU GENLIST TEMPLATE FILE (#90064.2)UCI: DEV,DEV
V
ERSION: 1.2

STORED IN: ^BGUPLATE(                                      10/01/01    PAGE 1
------------------------------------------------------------------------------
--

                                                  FILE SECURITY
                                   DD SECURITY    : @     DELETE SECURITY: @
                                   READ SECURITY  : @     LAYGO SECURITY : @
                                   WRITE SECURITY : @
CROSS REFERENCED BY:
      NAME(B)


                                 FILE STRUCTURE

FIELD     FIELD
NUMBER    NAME

.01       NAME (RF), [0;1]
.02       BGUFILE (P1'), [0;2]
.03       BGUIEN (NJ7,0), [0;3]
.04       BGUMORE (F), [0;4]
.05       BGUCRFS (F), [0;5]
.06       BGUMAX (NJ4,0), [0;6]
.07       BGUBEGIN (F), [0;7]
.08       BGUEND (F), [0;8]
.09       BGUDIR (F), [0;9]
.1        BGUID (F), [0;10]
1         BGUVLST (Multiple-90064.21), [1;0]
          .01  BGUVLST (WL), [0;1]
2         BGUSCR (F), [2;1]
3         BGUCNDS (F), [3;1]
4         REMOTE CALL (F), [4;1]



CONDENSED DATA DICTIONARY---BGU MUMPS CODE FILE (#90065)UCI: DEV,DEV
VERSION:
```

```
1.2

STORED IN: ^BGUMCD(                                    10/01/01    PAGE 1
------------------------------------------------------------------------------
--

                                                 FILE SECURITY
                                   DD SECURITY    : @     DELETE SECURITY: B
                                   READ SECURITY  : B     LAYGO SECURITY : B
                                   WRITE SECURITY : B
CROSS REFERENCED BY:
      NAME(B)  MUMPS CODE(C)


                                 FILE STRUCTURE

FIELD     FIELD
NUMBER    NAME

.01       NAME (RF), [0;1]
.02       MUMPS CODE (RF), [0;2]
.03       DATE (RD), [0;3]
.04       ACTIVE FLAG (RS), [0;4]
.05       DEVELOPER (RP200'), [0;5]
.06       DESCRIPTION (Multiple-90065.01), [1;0]
          .01  DESCRIPTION (W), [0;1]



CONDENSED DATA DICTIONARY---BGU PRESET REMOTE PROCEDURE CALL FILE (#90066)UCI:
D
EV,DEV                                                           VERSION: 1.2

STORED IN: ^BGUPSRPC(                                  10/01/01    PAGE 1
------------------------------------------------------------------------------
--

                                                 FILE SECURITY
                                   DD SECURITY    : @     DELETE SECURITY: B
                                   READ SECURITY  : B     LAYGO SECURITY : B
                                   WRITE SECURITY : B
CROSS REFERENCED BY:
      NAME(B)


                                 FILE STRUCTURE

FIELD     FIELD
NUMBER    NAME

.01       NAME (RF), [0;1]
.02       REMOTE PROCEDURE (RP8994'), [0;2]
.03       PARAMETER (Multiple-90066.01), [1;0]
          .01  PARAMETER NUMBER (MNJ2,0), [0;1]
          .02  VALUE (F), [0;2]



CONDENSED DATA DICTIONARY---BGU USER GUI PROFILE FILE (#90069)UCI: DEV,DEV
VER
```

```
SION: 1.2

STORED IN: ^BGUUSER(                                      10/01/01    PAGE 1
------------------------------------------------------------------------------
--

                                                  FILE SECURITY
                                   DD SECURITY    : @      DELETE SECURITY: B
                                   READ SECURITY  : B      LAYGO SECURITY : B
                                   WRITE SECURITY : B
CROSS REFERENCED BY:
      NAME(B)


                                 FILE STRUCTURE

FIELD     FIELD
NUMBER    NAME

.01       NAME (RP200'X), [0;1]
21        INSTITUTION (Multiple-90069.021), [21;0]
          .01  INSTITUTION (MP4'X), [0;1]
          .02  INBOUND CONNECTION ALLOWED (S), [0;2]
          .03  OUBOUND CONNECTION ALLOWED (S), [0;3]
          .04  IP ADDRESS (F), [0;4]
          .05  IP PORT NUMBER (SOCKET) (NJ5,0), [0;5]
          .06  ACCESS CODE (F), [0;6]
          .07  VERIFY CODE (F), [0;7]
          11   ALLOWED MAC ADDRESS (Multiple-90069.2111), [11;0]
               .01  ALLOWED MAC ADDRESS (MF), [0;1]
          21   BGU PACKAGE SUPPORTED (Multiple-90069.2121), [21;0]
               .01  BGU PACKAGE SUPPORTED (MP90069.1'X), [0;1]
               .02  PARAM02 (F), [0;2]
               .03  PARAM03 (F), [0;3]
               .04  PARAM04 (F), [0;4]
               .05  PARAM05 (F), [0;5]
               .06  PARAM06 (F), [0;6]
               .07  PARAM07 (F), [0;7]
               .08  PARAM08 (F), [0;8]
51        PACKAGE (Multiple-90069.051), [51;0]
          .01  PACKAGE (MP9.4'X), [0;1]
          .02  PARAM02 (F), [0;2]
          .03  PARAM03 (F), [0;3]
          .04  PARAM04 (F), [0;4]
          .05  PARAM05 (F), [0;5]
          .06  PARAM06 (F), [0;6]
          .07  PARAM07 (F), [0;7]
          .08  PARAM08 (F), [0;8]


Select DATA DICTIONARY UTILITY OPTION:
*****
*****

STANDARD DATA DICTIONARY #90061 -- BGU TRACE FILE              10/1/01  PAGE
1
STORED IN ^BGUTRACE(  (2362 ENTRIES)   SITE: CROW HOSPITAL   UCI: DEV,DEV
(VERSI
ON 1.2)
DATA          NAME                  GLOBAL        DATA
ELEMENT       TITLE                 LOCATION      TYPE
------------------------------------------------------------------------------
-



              DD ACCESS: @
              RD ACCESS: B
              WR ACCESS: B
             DEL ACCESS: B
           LAYGO ACCESS: B
           AUDIT ACCESS: @

IDENTIFIED BY: APPLICATION ID (#.02)

CROSS
REFERENCED BY: SEQUENCE NUMBER(B), $J(C), APPLICATION ID(D)


90061,.01     SEQUENCE NUMBER        0;1 NUMBER (Required)

              INPUT TRANSFORM:  K:+X'=X!(X>99999)!(X<1)!(X?.E1"."1N.N) X
S:$D(X
                                ) DINUM=X
              LAST EDITED:      FEB 09, 1996
              HELP-PROMPT:      Type a Number between 1 and 99999
              DESCRIPTION:      The next sequential number assigned to this
                                trace.

              NOTES:            XXXX--CAN'T BE ALTERED EXCEPT BY PROGRAMMER

              CROSS-REFERENCE:  90061^B
                                1)= S ^BGUTRACE("B",$E(X,1,30),DA)=""
                                2)= K ^BGUTRACE("B",$E(X,1,30),DA)


90061,.02     APPLICATION ID         0;2 FREE TEXT (Required)

              INPUT TRANSFORM:  K:$L(X)>10!($L(X)<5) X
              LAST EDITED:      FEB 06, 1996
              HELP-PROMPT:      Answer must be 5-10 characters in length.
              DESCRIPTION:      The ID of the application that called the
                                trace.

              CROSS-REFERENCE:  90061^D
                                1)= S ^BGUTRACE("D",$E(X,1,30),DA)=""
                                2)= K ^BGUTRACE("D",$E(X,1,30),DA)


90061,.03     $J                     0;3 NUMBER (Required)

              INPUT TRANSFORM:  K:+X'=X!(X>999999)!(X<1)!(X?.E1"."1N.N) X
              LAST EDITED:      FEB 06, 1996
              HELP-PROMPT:      Type a Number between 1 and 999999
              DESCRIPTION:
                                The $J for this partition.

              NOTES:            XXXX--CAN'T BE ALTERED EXCEPT BY PROGRAMMER

              CROSS-REFERENCE:  90061^C
```

```
                                1)= S ^BGUTRACE("C",$E(X,1,30),DA)=""
                                2)= K ^BGUTRACE("C",$E(X,1,30),DA)


90061,.04     DATE/TIME              0;4 DATE (Required)

              INPUT TRANSFORM:  S %DT="ESTR" D ^%DT S X=Y K:Y<1 X
              LAST EDITED:      FEB 05, 1996
              HELP-PROMPT:      Enter a fileman date and time
              DESCRIPTION:
                                The time the trace information was saved.

              TECHNICAL DESCR:
                                The time the trace information was saved.


90061,.05     BUFFER STRING          1;0   WORD-PROCESSING #90061.01


90061,.06     XQOR STRING            2;0   WORD-PROCESSING #90061.02


90061,.07     RETURN DATA            3;0   WORD-PROCESSING #90061.03


INPUT TEMPLATE(S):

PRINT TEMPLATE(S):

SORT TEMPLATE(S):

FORM(S)/BLOCK(S):

STANDARD DATA DICTIONARY #90062 -- BGU SITE PARAMETERS FILE     10/1/01  PAGE
1
STORED IN ^BGUSP(  (1 ENTRY)   SITE: CROW HOSPITAL   UCI: DEV,DEV (VERSION
1.2)


DATA          NAME                  GLOBAL        DATA
ELEMENT       TITLE                 LOCATION      TYPE
-------------------------------------------------------------------------------
-


              DD ACCESS: @
              RD ACCESS: B
              WR ACCESS: B
             DEL ACCESS: B
           LAYGO ACCESS: B
           AUDIT ACCESS: @


CROSS
REFERENCED BY: SITE(B)


90062,.01     SITE                   0;1 POINTER TO KERNEL SITE PARAMETERS
FILE
                                 (#4.3) (Required)
              LAST EDITED:      FEB 12, 1996
              HELP-PROMPT:      Enter a site code
              CROSS-REFERENCE:  90062^B
                                1)= S ^BGUSP("B",$E(X,1,30),DA)=""
                                2)= K ^BGUSP("B",$E(X,1,30),DA)


90062,.02     TRACE ALL              0;2 SET (Required)

                                'Y' FOR YES;
                                'N' FOR NO;
              LAST EDITED:      JUL 25, 2001
              HELP-PROMPT:      Answer yes or no
              DESCRIPTION:      Allows user to turn on trace function for all
                                users

              TECHNICAL DESCR:  This should be turned on when auditing remote
                                access


90062,.03     $J TO TRACE            1;0 Multiple #90062.01

              DESCRIPTION:
                                The job number of the TRACE

              TECHNICAL DESCR:  Enter the job number.  can be obtained from
the
                                system status


90062.01,.01    $J TO TRACE            0;1 NUMBER

                INPUT TRANSFORM:  K:+X'=X!(X>999999)!(X<1)!(X?.E1"."1N.N) X
                LAST EDITED:      FEB 12, 1996
                HELP-PROMPT:      Type a Number between 1 and 999999
                CROSS-REFERENCE:  90062.01^B
                                  1)= S ^BGUSP(DA(1),1,"B",$E(X,1,30),DA)=""
                                  2)= K ^BGUSP(DA(1),1,"B",$E(X,1,30),DA)


90062.01,1      Number of sessions to retain 0;2 NUMBER

                INPUT TRANSFORM:  K:+X'=X!(X>999)!(X<1)!(X?.E1"."1N.N) X
                LAST EDITED:      FEB 20, 1996
                HELP-PROMPT:      Type a Number between 1 and 999
                DESCRIPTION:      This controls the number of sessions
                                  (Sequence numbers) retained when file
cleanup
                                  is requested.



90062,.04     SOCKET                 2;1 NUMBER

              INPUT TRANSFORM:  K:+X'=X!(X>99999)!(X<1)!(X?.E1"."1N.N) X
              LAST EDITED:      MAR 05, 1996
              HELP-PROMPT:      Type a Number between 1 and 99999
              DESCRIPTION:      This is the socket that will be used by the
```

```
                                handler (BGUTCPH) when it is started in
                                ^SYSGEN.


90062,.05     STATISTICS             0;3 SET

                                'Y' FOR YES;
                                'N' FOR NO;
              LAST EDITED:      MAY 26, 1996
              HELP-PROMPT:      Enter yes or no
              DESCRIPTION:      Allows user to monitor GUI sign-on and number
                                and types of messages exercised.

              TECHNICAL DESCR:
                                stores the type of remote procedure invoked


90062,.06     SERVER TIMEOUT         0;6 NUMBER

              INPUT TRANSFORM:  K:+X'=X!(X>3600)!(X<300)!(X?.E1"."1N.N) X
              LAST EDITED:      MAR 29, 2001
              HELP-PROMPT:      Type a Number between 300 and 3600
              DESCRIPTION:      This is the time in seconds that the IHS Mumps
                                server will shutdown in if there is no
                                activity.  If this parameter is not set then
                                the server will timeout in one hour (3600
                                seconds).


90062,.07     DAYS TO KEEP ENTRIES   0;7 NUMBER

              INPUT TRANSFORM:  K:+X'=X!(X>120)!(X<0)!(X?.E1"."1N.N) X
              LAST EDITED:      MAY 07, 2001
              HELP-PROMPT:      Type a Number between 0 and 120
              DESCRIPTION:      Controls the number of days to keep entries in
                                ^BGUTRACE.  It will use the 4th piece of the
                                zero node as the reference date.



      FILES POINTED TO                    FIELDS

KERNEL SITE PARAMETERS (#4.3)     SITE (#.01)


INPUT TEMPLATE(S):

PRINT TEMPLATE(S):

SORT TEMPLATE(S):

FORM(S)/BLOCK(S):

STANDARD DATA DICTIONARY #90062.1 -- BGU SIGN-ON STATISTICS FILE   10/1/01
PAGE
 1
STORED IN ^BGUSEC(  (185 ENTRIES)   SITE: CROW HOSPITAL   UCI: DEV,DEV
(VERSION
1.2)
DATA          NAME                  GLOBAL        DATA
ELEMENT       TITLE                 LOCATION      TYPE
------------------------------------------------------------------------------
-
This file records the GUI user statistics when the BGU STATISTICS SITE
PARAMETERS is set to Yes.  This file records the sign-on/sign-off times by
user.  A statistical count of all calls is recorderd by call.  Additional data
about the connection is recorded.


              DD ACCESS: @
              RD ACCESS: B
              WR ACCESS: B
             DEL ACCESS: B
           LAYGO ACCESS: B
           AUDIT ACCESS: @


CROSS
REFERENCED BY: SIGNOFF TIME(AC), USER(CUR)


90062.1,.001  DATE/TIME                  DATE

              INPUT TRANSFORM:  S %DT="ESTXR" D ^%DT S X=Y K:Y<1 X
              LAST EDITED:      JAN 30, 1992
              HELP-PROMPT:      Enter date and time.
              DESCRIPTION:
                                This is the date/time that the user signed on.


90062.1,.01   USER                   0;1 POINTER TO NEW PERSON FILE (#200)
                                 (Required)

              LAST EDITED:      JUN 18, 1991
              DESCRIPTION:
                                This is the user that signed on.

              SOURCE OF DATA:   SIGN-ON SYSTEM
              CROSS-REFERENCE:  90062.1^CUR
                                1)= S ^BGUSEC("CUR",$E(X,1,30),DA)=""
                                2)= K ^BGUSEC("CUR",$E(X,1,30),DA)


90062.1,1     DEVICE $I              0;2 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>15!($L(X)<1) X
              HELP-PROMPT:      $I VALUE AT SIGN-ON
              DESCRIPTION:
                                This is the device that the user signed on to.


90062.1,2     JOB                    0;3 NUMBER

              INPUT TRANSFORM:  K:+X'=X!(X>9999999)!(X<0)!(X?.E1"."1N.N) X
              HELP-PROMPT:      $J (PARTITION NUMBER) OF SIGN-ON
              DESCRIPTION:
                                This is the job number of the user's job.
```

```
90062.1,3     SIGNOFF TIME          0;4 DATE

              INPUT TRANSFORM:  S %DT="ESTXR" D ^%DT S X=Y K:Y<1 X
              LAST EDITED:      JAN 30, 1992
              HELP-PROMPT:      Enter Date and Time
              DESCRIPTION:      This is the time that the user signed-off
                                through H^XUS.

              CROSS-REFERENCE:  90062.1^AC^MUMPS
                                1)= K ^BGUSEC("CUR",+^BGUSEC(DA,0),DA)
                                2)= Q
                                3)= The CUR x-ref.
                                This X-ref clears ^BGUSEC('CUR' that keeps a
                                list of user sign-ons  that is used by the
FIND
                                USER option.


90062.1,4     NODE NAME             0;5 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>50!($L(X)<1) X
              LAST EDITED:      MAY 26, 1996
              HELP-PROMPT:      Answer must be 1-50 characters in length.
              DESCRIPTION:
                                This is the IP name of the user signed on.

              TECHNICAL DESCR:
                                This parameter is passed from the BGUSYNC OCX.


90062.1,5     SOCKET                0;7 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>8!($L(X)<1) X
              LAST EDITED:      MAY 26, 1996
              HELP-PROMPT:      Answer must be 1-8 characters in length.
              DESCRIPTION:      This is the socket number passed from the
                                BGUSYNC OCX to the server.  The socket is
                                unique to the session.

              TECHNICAL DESCR:  The socket is established from a port address
                                and IP address of the client and IP address
and
                                port of the server.  This pair is unique to
the
                                entire world and assures that the partition is
                                a unique user.


90062.1,6     ETHERNET ADDRESS      0;8 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>18!($L(X)<1) X
              LAST EDITED:      MAY 26, 1996
              HELP-PROMPT:      Answer must be 1-18 characters in length.
              DESCRIPTION:      This is the physical ethernet address of the
                                client and is passed from the BGUSYNC OCX at
                                the security call.

              TECHNICAL DESCR:  The physical ethernet card is unique to the
                                world and is set by the vendor at time of
                                manufacture.
```

```
90062.1,7     APPLICATION ID         0;6 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>30!($L(X)<3) X
              LAST EDITED:      MAY 26, 1996
              HELP-PROMPT:      Answer must be 3-30 characters in length.
              DESCRIPTION:      The client application should call the BGUSYNC
                                OCX property IHSApplicationID to pass the
                                current application ID and version number.

              TECHNICAL DESCR:  If the client application does not call the
                                BGUSYNC OCX property, there will be no data
                                passed.


90062.1,10    PARAMETER              0;10 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>50!($L(X)<1) X
              LAST EDITED:      MAY 26, 1996
              HELP-PROMPT:      Answer must be 1-50 characters in length.
              DESCRIPTION:      This is the SecurityParam passed from the
                                BGUSYNC OCX.  It is a special free text param
                                that is reserved for future use.

              TECHNICAL DESCR:  This is created by calling the BGUSYNC OCX
                                property SecurityParam and is additional data
                                passed by the client at the security sign-on.


90062.1,20    CALL STATISTICS        20;0 Multiple #90062.11
                                 (Add New Entry without Asking)

              DESCRIPTION:      This is a multiple and contains each of the
                                calls made during the session.  These message
                                headers are found in the REMOTE PROCEDURE
FILE.


              TECHNICAL DESCR:  This field is created if the BGU SITE
PARAMETER
                                field SECURITY is answered Yes.


90062.11,.01    CALL STATISTICS        0;1 FREE TEXT (Multiply asked)

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<3) X
                LAST EDITED:      MAY 26, 1996
                HELP-PROMPT:      Answer must be 3-30 characters in length.
                DESCRIPTION:      Tracks each call and the number of times the
                                  user made the call.

                CROSS-REFERENCE:  90062.11^B
                                  1)= S ^BGUSEC(DA(1),20,"B",$E(X,1,30),DA)=""
                                  2)= K ^BGUSEC(DA(1),20,"B",$E(X,1,30),DA)


90062.11,.02    NUMBER                 0;2 NUMBER

                INPUT TRANSFORM:  K:+X'=X!(X>999)!(X<0)!(X?.E1"."1N.N) X
                LAST EDITED:      MAY 26, 1996
                HELP-PROMPT:      Type a Number between 0 and 999, 0 Decimal
                                  Digits
                DESCRIPTION:      Total number of calls for this message
header
                                  during the session.  Used for useage
                                  statistics.




90062.1,97    ELAPSED TIME (SECONDS)  ;  COMPUTED

              MUMPS CODE:       X ^DD(90062.1,97,9.2) S X1=Y(90062.1,97,1) S
X=
                                $S(X&X1:X,1:"") Q:'X  D H^%DTC S
XUS1=%H*86400+
                                %T,X=X1 D H^%DTC S
XUS2=%H*86400+%T,X=$S(XUS1>X
                                US2:XUS1-XUS2,1:XUS2-XUS1)
                                9.2 = S
Y(90062.1,97,2)=$S($D(^BGUSEC(D0,0)):^(
                                0),1:"") S
X=$S('$D(D0):"",D0<0:"",1:D0),Y(9006
                                2.1,97,1)=X S X=$P(Y(90062.1,97,2),U,4)
              ALGORITHM:        XUDELTATIME(#.001,#3)
              DESCRIPTION:      This field calculates the number of seconds
                                between sign-on time and sign-off time.


90062.1,99    ELAPSED TIME (MINUTES)  ;  COMPUTED

              MUMPS CODE:       S X1=$P(^BGUSEC(D0,0),U,4),X="" Q:X1<2000000
S
                                 X=D0,Y=$E(X1_"000",9,10)-
$E(X_"000",9,10)*60+$
                                E(X1_"00000",11,12)-
$E(X_"00000",11,12),X2=X,X=
                                $P(X,".",1)'=$P(X1,".",1) D ^%DTC:X S
X=X*1440+
                                Y
              ALGORITHM:        MINUTES(#3,#.001)
              DESCRIPTION:      This is the amount of time the user has been
                                signed on.

              NOTES:            XXXX--CAN'T BE ALTERED EXCEPT BY PROGRAMMER



      FILES POINTED TO                      FIELDS

NEW PERSON (#200)                 USER (#.01)


INPUT TEMPLATE(S):

PRINT TEMPLATE(S):

SORT TEMPLATE(S):
BGU-SIGNON-USER               JUN 18, 1996@14:32  USER #0
```

```
SORT BY: USER// (USER not null)
  WITHIN USER, SORT BY: @DATE/TIME// (DATE/TIME not null)


FORM(S)/BLOCK(S):

STANDARD DATA DICTIONARY #90063 -- BGU SERVER FILE              10/1/01  PAGE
1
STORED IN ^BGUSV(  *** NO DATA STORED YET ***   SITE: CROW HOSPITAL   UCI:
DEV,D
EV                                                             (VERSION 1.2)

DATA           NAME                  GLOBAL        DATA
ELEMENT        TITLE                 LOCATION      TYPE
-------------------------------------------------------------------------------
-



              DD ACCESS: @
              RD ACCESS: B
              WR ACCESS: B
             DEL ACCESS: B
           LAYGO ACCESS: B
           AUDIT ACCESS: @


CROSS
REFERENCED BY: NAME(B)


90063,.01      NAME                   0;1 FREE TEXT (Required)

               INPUT TRANSFORM:  K:$L(X)>30!($L(X)<3)!'(X'?1P.E) X
               LAST EDITED:      APR 25, 1995
               HELP-PROMPT:      Answer must be 3-30 characters in length.
               CROSS-REFERENCE:  90063^B
                                 1)= S ^BGUSV("B",$E(X,1,30),DA)=""
                                 2)= K ^BGUSV("B",$E(X,1,30),DA)


90063,1        PORT                   0;2 FREE TEXT

               INPUT TRANSFORM:  K:$L(X)>6!($L(X)<1) X
               LAST EDITED:      APR 25, 1995
               HELP-PROMPT:      Answer must be 1-6 characters in length.

INPUT TEMPLATE(S):

PRINT TEMPLATE(S):

SORT TEMPLATE(S):

FORM(S)/BLOCK(S):

STANDARD DATA DICTIONARY #90064 -- BGU TABLE FILE               10/1/01  PAGE
1
STORED IN ^BGUT(  *** NO DATA STORED YET ***   SITE: CROW HOSPITAL   UCI:
DEV,DE
V                                                              (VERSION 1.2)
```

```
DATA          NAME                  GLOBAL        DATA
ELEMENT       TITLE                 LOCATION      TYPE
------------------------------------------------------------------------------
-


              DD ACCESS: @
              RD ACCESS: B
              WR ACCESS: B
             DEL ACCESS: B
           LAYGO ACCESS: B
           AUDIT ACCESS: @


CROSS
REFERENCED BY: NAME(B)


90064,.01     NAME                   0;1 FREE TEXT (Required)

              INPUT TRANSFORM:  K:$L(X)>30!(X?.N)!($L(X)<3)!'(X'?1P.E) X
              HELP-PROMPT:      NAME MUST BE 3-30 CHARACTERS, NOT NUMERIC OR
                                STARTING WITH PUNCTUATION
              CROSS-REFERENCE:  90064^B
                                1)= S ^BGUT("B",$E(X,1,30),DA)=""
                                2)= K ^BGUT("B",$E(X,1,30),DA)


90064,1       FILE NUMBER            0;2 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
              LAST EDITED:      MAR 07, 1995
              HELP-PROMPT:      Answer must be 1-30 characters in length.

90064,2       FIELD                  1;0 Multiple #90064.02


90064.02,.01    FIELD                  0;1 FREE TEXT (Multiply asked)

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<3) X
                LAST EDITED:      FEB 23, 1995
                HELP-PROMPT:      Answer must be 3-30 characters in length.
                CROSS-REFERENCE:  90064.02^B
                                  1)= S ^BGUT(DA(1),1,"B",$E(X,1,30),DA)=""
                                  2)= K ^BGUT(DA(1),1,"B",$E(X,1,30),DA)


90064.02,1      FIELD NUMBER           0;2 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                LAST EDITED:      MAR 07, 1995
                HELP-PROMPT:      Answer must be 1-30 characters in length.



INPUT TEMPLATE(S):

PRINT TEMPLATE(S):

SORT TEMPLATE(S):
```

```
FORM(S)/BLOCK(S):

STANDARD DATA DICTIONARY #90064.1 -- BGU EXECUTE CODE FILE      10/1/01  PAGE
1
STORED IN ^BGUEXC(  *** NO DATA STORED YET ***   SITE: CROW HOSPITAL   UCI:
DEV,
DEV                                                               (VERSION 1.2)

DATA           NAME                   GLOBAL        DATA
ELEMENT        TITLE                  LOCATION      TYPE
-------------------------------------------------------------------------------
-
The execute code file is used to store a variety of program instuctions that
are used in various programs in the lab package.  The best way to see what
they
areand what they do is to sort them out by type.  This may only be of academic
interest, since these codes are rarely changed.

The 'execute code' type entries have been moved from this file to the routine
LRX.  All of the 'execute code' type entries will be deleted (with the
exception of the ERROR TRAP entry) when version 5 of lab is released.

Cross-reference description:

 ^LAB("X",'SUBSCRIPT NAME')=EXECUTE CODE

   Note: With the exception of the Error Trap execute code this x-ref has
         been replaced with the program LRX.
         Example:  X ^LAB("X","PT") is now D PT^LRX


              DD ACCESS: @
              RD ACCESS: B
              WR ACCESS: B
             DEL ACCESS: B
           LAYGO ACCESS: B
           AUDIT ACCESS: @


CROSS
REFERENCED BY: NAME(B), EXECUTE CODE(C), ACTIVATION(D)


90064.1,.01    NAME                    0;1 FREE TEXT (Required)

               INPUT TRANSFORM:  K:$L(X)>30!($L(X)<2)!'(X'?1P.E)!(X'?.ANP) X
               LAST EDITED:      NOV 28, 1984
               HELP-PROMPT:      ANSWER MUST BE 2-30 CHARACTERS IN LENGTH
               DESCRIPTION:
                                 The user given name of the code.

               CROSS-REFERENCE:  90064.1^B
                                 1)= S ^BGUEXC("B",$E(X,1,30),DA)=""
                                 2)= K ^BGUEXC("B",$E(X,1,30),DA)


90064.1,1      EXECUTE CODE            .1;E1,220 FREE TEXT

               INPUT TRANSFORM:  D ^DIM
               LAST EDITED:      OCT 25, 1984
               HELP-PROMPT:      ANSWER MUST BE 1-220 CHARACTERS IN LENGTH. AND
                                 VALID MUMPS CODE
               DESCRIPTION:      The MUMPS code.  EXAMPLE:  For the SMAC name
                                 the code is S CUP=CUP+1 I CUP>8 S
                                 CUP=1,TRAY=TRAY+1.

               NOTES:            XXXX--CAN'T BE ALTERED EXCEPT BY PROGRAMMER

               CROSS-REFERENCE:  90064.1^C
                                 1)= S ^BGUEXC("C",$E(X,1,30),DA)=""
                                 2)= K ^BGUEXC("C",$E(X,1,30),DA)
                                 The 'C' cross reference contains the
executable
                                 code for quick   pattern match.


90064.1,2     ACTIVATION             0;2 SET

                                 'A' FOR ACTIVATED;
                                 'I' FOR INACTIVATED;
               LAST EDITED:      AUG 20, 1996
               HELP-PROMPT:      Enter 'A' if this is Activated code, enter 'I'
                                 for inactivated.
               DESCRIPTION:      This field sets the flag to determine if this
                                 execute code can be used by the message
                                 handler.  If the field is inactive, the code
is
                                 out of service and will not be executed.  If
                                 the field is active, the code will be executed
                                 by the message call.

               TECHNICAL DESCR:  All executable code passed to the server for
                                 RPC and API calls must exist in this file.  A
                                 pattern match is required for the code to be
                                 executed.  This process prevents unauthorized
                                 executable mumps code from being processed by
                                 an application with out verification from the
                                 server.

               CROSS-REFERENCE:  90064.1^D
                                 1)= S ^BGUEXC("D",$E(X,1,30),DA)=""
                                 2)= K ^BGUEXC("D",$E(X,1,30),DA)
                                 Creates cross-reference to verify if call is
                                 activated.


90064.1,3     TYPE                   0;3 SET

                                 'A' FOR API;
                                 'R' FOR RPC;
               LAST EDITED:      AUG 20, 1996
               HELP-PROMPT:      Enter an 'A' for API, 'R' for RPC or leave
                                 blank.
               DESCRIPTION:      The type of code is categorized by the set of
                                 codes.  The entry can either be A for API, R
                                 for RPC or blank if any message call can use
                                 this execute code.
               TECHNICAL DESCR:  The type field determines the message type
 that
                                 is using this execute code.  If the field
                                 contains an 'A' then the APICall or APITable
                                 BGU Remote Procedure Call can use this
                                 executable code if it is activated.  If the
                                 field contains an 'R' then the RPCCall or
                                 RPCTable BGU Remote Procedure Call can use
 this
                                 executable code if it is activated.  If the
                                 field is blank, than any call can use this
 code
                                 if activated.


 90064.1,4     ERROR PROMPT           .2;1 FREE TEXT

               INPUT TRANSFORM:  K:$L(X)>200!($L(X)<2) X
               LAST EDITED:      AUG 28, 1984
               HELP-PROMPT:      Enter the returned message description.
               DESCRIPTION:      The error return message if the code cannot be
                                 used.

               TECHNICAL DESCR:  Contains the string that is passed back with
                                 the error value (usually a -1 count) telling
                                 the user that the code cannot be executed.  If
                                 the code pattern matches the code sent in the
                                 message, but the code is inactivated, than
 this
                                 message is passed.


 90064.1,5     APPLICATION NAME       2;1 POINTER TO PACKAGE FILE (#9.4)

               LAST EDITED:      AUG 20, 1996
               HELP-PROMPT:      Enter the package name that uses this
                                 executable call.
               DESCRIPTION:      This is the package name that is using this
                                 call.


 90064.1,6     DESCRIPTION            1;0   WORD-PROCESSING #90064.101

               DESCRIPTION:      This field contains information about how the
                                 execute code is used, or what it does.

               TECHNICAL DESCR:  This is a word processing field where the
                                 developer will describe the use of this
                                 executable code.



       FILES POINTED TO                      FIELDS

 PACKAGE (#9.4)                    APPLICATION NAME (#5)


 INPUT TEMPLATE(S):

 PRINT TEMPLATE(S):
```

```
SORT TEMPLATE(S):

FORM(S)/BLOCK(S):

STANDARD DATA DICTIONARY #90064.2 -- BGU GENLIST TEMPLATE FILE   10/1/01  PAGE
1
STORED IN ^BGUPLATE(  (9 ENTRIES)   SITE: CROW HOSPITAL   UCI: DEV,DEV
(VERSION
1.2)

DATA          NAME                  GLOBAL        DATA
ELEMENT       TITLE                 LOCATION      TYPE
-------------------------------------------------------------------------------
-



              DD ACCESS: @
              RD ACCESS: @
              WR ACCESS: @
             DEL ACCESS: @
           LAYGO ACCESS: @
           AUDIT ACCESS: @



CROSS
REFERENCED BY: NAME(B)


90064.2,.01   NAME                   0;1 FREE TEXT (Required)

              INPUT TRANSFORM:  K:$L(X)>30!($L(X)<3)!'(X'?1P.E) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Answer must be 3-30 characters in length.
              DESCRIPTION:      This is the record set name used in testing
BGU
                                GENLIST on the server.

              CROSS-REFERENCE:  90064.2^B
                                1)= S ^BGUPLATE("B",$E(X,1,30),DA)=""
                                2)= K ^BGUPLATE("B",$E(X,1,30),DA)


90064.2,.02   BGUFILE                0;2 POINTER TO FILE FILE (#1)

              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Enter file name or number
              DESCRIPTION:
                                Allows user to select file.

              TECHNICAL DESCR:  Primary file providing the data.  e.g. 2 for
                                DPT The entry can be either the file name or
                                number.  The number is stored because this is
a
                                pointer to the FILE file (1)


90064.2,.03   BGUIEN                 0;3 NUMBER

              INPUT TRANSFORM:  K:+X'=X!(X>9999999)!(X<0)!(X?.E1"."1N.N) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Type a Number between 0 and 9999999, 0 Decimal
                                Digits
              DESCRIPTION:
                                Allows user to specify the IEN of the file.

              TECHNICAL DESCR:  Internal Entry Number - If this is set, fields
                                specific only to this record are returned.


90064.2,.04   BGUMORE                0;4 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>1!($L(X)<1) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Answer must be 1 character in length.
              DESCRIPTION:      Sets the MORE button to 1 (one) to continue
                                with record sets.

              TECHNICAL DESCR:  Flag is set on (=1) if you want the program to
                                start from where it left off on the previous
                                call/  If this flag isn't set then the program
                                will start at the beginning determined by the
                                value of BGUBEGIN


90064.2,.05   BGUCRFS                0;5 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Answer must be 1-30 characters in length.
              DESCRIPTION:      Allows user to enter a cross reference used in
                                lookup

              TECHNICAL DESCR:  THe cross-reference that is used to access the
                                main file.  You can use special lookup cross
                                references i.e. SC:TAG|ROUTINE|


90064.2,.06   BGUMAX                 0;6 NUMBER

              INPUT TRANSFORM:  K:+X'=X!(X>9999)!(X<1)!(X?.E1"."1N.N) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Type a Number between 1 and 9999, 0 Decimal
                                Digits
              DESCRIPTION:      Allows user to define the max number of record
                                sets to return.  The default is 25


90064.2,.07   BGUBEGIN               0;7 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Answer must be 1-30 characters in length.
              DESCRIPTION:      The starting position in the cross-reference
                                file if one is specified, otherwise the
                                starting point in the main file.  This
                                parameter is only used when in the list mode.


90064.2,.08   BGUEND                 0;8 FREE TEXT
```

```
              INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Answer must be 1-30 characters in length.
              TECHNICAL DESCR:
                                The opposite of BGUBEGIN


90064.2,.09   BGUDIR                 0;9 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>1!($L(X)<1) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Answer must be 1 character in length.
              DESCRIPTION:      Sets the direction the program is traversing
                                through the file.  F for forward and B for
                                backward.  The default is F


90064.2,.1    BGUID                  0;10 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Answer must be 1-30 characters in length.
              DESCRIPTION:      Identification parameter - the IP address plus
                                time including seconds.


90064.2,1     BGUVLST                1;0   WORD-PROCESSING #90064.21 (NOWRAP)


90064.2,2     BGUSCR                 2;1 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>240!($L(X)<1) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Answer must be 1-240 characters in length.
              DESCRIPTION:      Allows user to set MUMPS executable code for
                                screening valid data.  If a screen is defined,
                                only data that passes the screen is returned.
                                e.g.  BGUV(BGUFILE,.03)<1  tests to determine
                                if field .03 of the main file contains a value
                                of less than 1 or null and if so will accept
as
                                having passed the screen.


90064.2,3     BGUCNDS                3;1 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>240!($L(X)<1) X
              LAST EDITED:      MAR 02, 2000
              HELP-PROMPT:      Answer must be 1-240 characters in length.
              DESCRIPTION:      Englis like condition for screening data e.g.
                                SEX .EQ. "M" AND AGE > 40


90064.2,4     REMOTE CALL            4;1 FREE TEXT

              INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
              LAST EDITED:      APR 20, 2000
              HELP-PROMPT:      Answer must be 1-30 characters in length.
```

```
      FILES POINTED TO                      FIELDS

FILE (#1)                          BGUFILE (#.02)


INPUT TEMPLATE(S):

PRINT TEMPLATE(S):

SORT TEMPLATE(S):

FORM(S)/BLOCK(S):

STANDARD DATA DICTIONARY #90065 -- BGU MUMPS CODE FILE          10/1/01  PAGE
1
STORED IN ^BGUMCD(  *** NO DATA STORED YET ***   SITE: CROW HOSPITAL   UCI:
DEV,
DEV                                                            (VERSION 1.2)

DATA          NAME                  GLOBAL        DATA
ELEMENT       TITLE                 LOCATION      TYPE
-------------------------------------------------------------------------------
-


              DD ACCESS: @
              RD ACCESS: B
              WR ACCESS: B
             DEL ACCESS: B
           LAYGO ACCESS: B
           AUDIT ACCESS: @


CROSS
REFERENCED BY: NAME(B), MUMPS CODE(C)


90065,.01     NAME                   0;1 FREE TEXT (Required)

              INPUT TRANSFORM:  K:$L(X)>80!($L(X)<3)!'(X'?1P.E) X
              LAST EDITED:      SEP 13, 1996
              HELP-PROMPT:      Answer must be 3-80 characters in length.
              CROSS-REFERENCE:  90065^B
                                1)= S ^BGUMCD("B",$E(X,1,30),DA)=""
                                2)= K ^BGUMCD("B",$E(X,1,30),DA)


90065,.02     MUMPS CODE             0;2 FREE TEXT (Required)

              INPUT TRANSFORM:  K:$L(X)>100!($L(X)<1) X
              LAST EDITED:      SEP 13, 1996
              HELP-PROMPT:      Answer must be 1-100 characters in length.
              CROSS-REFERENCE:  90065^C
                                1)= S ^BGUMCD("C",$E(X,1,30),DA)=""
                                2)= K ^BGUMCD("C",$E(X,1,30),DA)


90065,.03     DATE                   0;3 DATE (Required)
```

```
              INPUT TRANSFORM:  S %DT="E" D ^%DT S X=Y K:Y<1 X
              LAST EDITED:      SEP 13, 1996
              HELP-PROMPT:      Enter Date

90065,.04     ACTIVE FLAG            0;4 SET (Required)

                                'A' FOR ACTIVE;
                                'I' FOR INACTIVE;
              LAST EDITED:      SEP 13, 1996
              HELP-PROMPT:      Enter A or I
              DESCRIPTION:
                                Determines if M code is active in system.

              TECHNICAL DESCR:  If set to NO, the M code will not be validated
                                as acceptable code.


90065,.05     DEVELOPER              0;5 POINTER TO NEW PERSON FILE (#200)
                                 (Required)

              LAST EDITED:      SEP 13, 1996

90065,.06     DESCRIPTION            1;0   WORD-PROCESSING #90065.01



      FILES POINTED TO                      FIELDS

NEW PERSON (#200)                 DEVELOPER (#.05)


INPUT TEMPLATE(S):

PRINT TEMPLATE(S):

SORT TEMPLATE(S):

FORM(S)/BLOCK(S):

STANDARD DATA DICTIONARY #90066 -- BGU PRESET REMOTE PROCEDURE CALL FILE
10/1/
01  PAGE 1
STORED IN ^BGUPSRPC(  (3 ENTRIES)   SITE: CROW HOSPITAL   UCI: DEV,DEV
(VERSION
1.2)

DATA          NAME                  GLOBAL        DATA
ELEMENT       TITLE                 LOCATION      TYPE
------------------------------------------------------------------------------
-



              DD ACCESS: @
              RD ACCESS: B
              WR ACCESS: B
             DEL ACCESS: B
           LAYGO ACCESS: B
           AUDIT ACCESS: @
CROSS
REFERENCED BY: NAME(B)


90066,.01     NAME                   0;1 FREE TEXT (Required)

              INPUT TRANSFORM:  K:$L(X)>30!($L(X)<3)!'(X'?1P.E) X
              LAST EDITED:      APR 09, 1997
              HELP-PROMPT:      Answer must be 3-30 characters in length.
              DESCRIPTION:
                                The name of the Preset Remote Procedure.

              TECHNICAL DESCR:  A remote procdure call that has all of its
                                parameters preset.

              CROSS-REFERENCE:  90066^B
                                1)= S ^BGUPSRPC("B",$E(X,1,30),DA)=""
                                2)= K ^BGUPSRPC("B",$E(X,1,30),DA)


90066,.02     REMOTE PROCEDURE       0;2 POINTER TO REMOTE PROCEDURE FILE
(#899
                                4) (Required)

              LAST EDITED:      APR 09, 1997
              DESCRIPTION:      Enter the remote procedure that uses the set
of
                                preset parameters.

              TECHNICAL DESCR:  The remote procedure that is called when this
                                definition is used.


90066,.03     PARAMETER              1;0 Multiple #90066.01


90066.01,.01    PARAMETER NUMBER       0;1 NUMBER (Multiply asked)

                INPUT TRANSFORM:  K:+X'=X!(X>99)!(X<1)!(X?.E1"."1N.N) X
                LAST EDITED:      APR 09, 1997
                HELP-PROMPT:      Type a Number between 1 and 99, 0 Decimal
                                  Digits
                DESCRIPTION:      The parameter order number used by the
remote
                                  procedure.

                TECHNICAL DESCR:  The sequence that this parameter appears in
                                  the parameter list for this procedure.

                CROSS-REFERENCE:  90066.01^B
                                  1)= S
^BGUPSRPC(DA(1),1,"B",$E(X,1,30),DA)=""
                                  2)= K ^BGUPSRPC(DA(1),1,"B",$E(X,1,30),DA)


90066.01,.02    VALUE                  0;2 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>200!($L(X)<1) X
                LAST EDITED:      APR 09, 1997
                HELP-PROMPT:      Answer must be 1-200 characters in length.
                DESCRIPTION:
                                  The value of the passed paramter.




      FILES POINTED TO                            FIELDS

REMOTE PROCEDURE (#8994)          REMOTE PROCEDURE (#.02)


INPUT TEMPLATE(S):

PRINT TEMPLATE(S):

SORT TEMPLATE(S):
MIKE                          APR 10, 2001@16:57  USER #3
SORT BY: NAME// (NAME from BPC to BPCZ)


FORM(S)/BLOCK(S):

STANDARD DATA DICTIONARY #90069 -- BGU USER GUI PROFILE FILE    10/1/01  PAGE
1
STORED IN ^BGUUSER(  *** NO DATA STORED YET ***   SITE: CROW HOSPITAL   UCI:
DEV
,DEV                                                           (VERSION 1.2)

DATA          NAME                  GLOBAL        DATA
ELEMENT       TITLE                 LOCATION      TYPE
------------------------------------------------------------------------------
-


              DD ACCESS: @
              RD ACCESS: B
              WR ACCESS: B
             DEL ACCESS: B
           LAYGO ACCESS: B
           AUDIT ACCESS: @


CROSS
REFERENCED BY: NAME(B)


90069,.01     NAME                   0;1 POINTER TO NEW PERSON FILE (#200)
                                 (Required)

              INPUT TRANSFORM:  S DINUM=X
              LAST EDITED:      FEB 17, 2000
              DESCRIPTION:
                                Profile of Graphical User Interface

              TECHNICAL DESCR:
                                Points to the VA(200 file

              NOTES:            XXXX--CAN'T BE ALTERED EXCEPT BY PROGRAMMER
              CROSS-REFERENCE:  90069^B
                                1)= S ^BGUUSER("B",$E(X,1,30),DA)=""
                                2)= K ^BGUUSER("B",$E(X,1,30),DA)


90069,21      INSTITUTION            21;0 POINTER Multiple #90069.021


90069.021,.01   INSTITUTION            0;1 POINTER TO INSTITUTION FILE (#4)
                                   (Multiply asked)

                INPUT TRANSFORM:  S DINUM=X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Enter GUI Institution
                DESCRIPTION:      Identifies facilities where this user can
                                  obtain patient data.

                NOTES:            XXXX--CAN'T BE ALTERED EXCEPT BY PROGRAMMER

                CROSS-REFERENCE:  90069.021^B
                                  1)= S
^BGUUSER(DA(1),21,"B",$E(X,1,30),DA)=""
                                  2)= K ^BGUUSER(DA(1),21,"B",$E(X,1,30),DA)


90069.021,.02   INBOUND CONNECTION ALLOWED 0;2 SET

                                  '0' FOR NO;
                                  '1' FOR YES;
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Enter Yes or No
                DESCRIPTION:      Determines if user is allowed to connect to
                                  this site

                TECHNICAL DESCR:  Controls access for user at this site for
                                  conections that originated at another
                                  facility.


90069.021,.03   OUBOUND CONNECTION ALLOWED 0;3 SET

                                  '0' FOR NO;
                                  '1' FOR YES;
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Enter Yes or No
                DESCRIPTION:      Determines if user is allowed to connect to
                                  anothersite

                TECHNICAL DESCR:  Controls access for user at another site for
                                  conections that originated at this facility.


90069.021,.04   IP ADDRESS             0;4 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>15!($L(X)<1) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 1-15 characters in length.

90069.021,.05   IP PORT NUMBER (SOCKET) 0;5 NUMBER
```

```
                INPUT TRANSFORM:  K:+X'=X!(X>65000)!(X<1)!(X?.E1"."1N.N) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Type a Number between 1 and 65000, 0 Decimal
                                  Digits

90069.021,.06   ACCESS CODE            0;6 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<6) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 6-30 characters in length.

90069.021,.07   VERIFY CODE            0;7 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<6) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 6-30 characters in length.

90069.021,11    ALLOWED MAC ADDRESS    11;0 Multiple #90069.2111


90069.2111,.01    ALLOWED MAC ADDRESS    0;1 FREE TEXT (Multiply asked)

                  INPUT TRANSFORM:  K:$L(X)>11!($L(X)<11) X
                  LAST EDITED:      FEB 17, 2000
                  HELP-PROMPT:      Answer must be 11 characters in length.
                  DESCRIPTION:      Contains MAC Ethernet address of
acceptable
                                    machines for logon

                  CROSS-REFERENCE:  90069.2111^B
                                    1)= S
^BGUUSER(DA(2),21,DA(1),11,"B",$E(X,1
                                    ,30),DA)=""

                                    2)= K
^BGUUSER(DA(2),21,DA(1),11,"B",$E(X,1
                                    ,30),DA)




90069.021,21    BGU PACKAGE SUPPORTED    21;0 POINTER Multiple #90069.2121


90069.2121,.01    BGU PACKAGE SUPPORTED    0;1 POINTER
                                       ** TO AN UNDEFINED FILE **
                                       (Multiply asked)

                    INPUT TRANSFORM:  S DINUM=X
                    LAST EDITED:      MAR 21, 2000
                    DESCRIPTION:      Identifies which GUI supported packages
                                      at remote site are supported

                    NOTES:            XXXX--CAN'T BE ALTERED EXCEPT BY
PROGRAMME
R

                    CROSS-REFERENCE:  90069.2121^B
                                      1)= S
^BGUUSER(DA(2),21,DA(1),21,"B",$E(X
                                      ,1,30),DA)=""

                                      2)= K
^BGUUSER(DA(2),21,DA(1),21,"B",$E(X
                                      ,1,30),DA)


90069.2121,.02    PARAM02                  0;2 FREE TEXT

                    INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                    LAST EDITED:      MAR 21, 2000
                    HELP-PROMPT:      Answer must be 1-30 characters in
length.

90069.2121,.03    PARAM03                  0;3 FREE TEXT

                    INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                    LAST EDITED:      MAR 21, 2000
                    HELP-PROMPT:      Answer must be 1-30 characters in
length.

90069.2121,.04    PARAM04                  0;4 FREE TEXT

                    INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                    LAST EDITED:      MAR 21, 2000
                    HELP-PROMPT:      Answer must be 1-30 characters in
length.

90069.2121,.05    PARAM05                  0;5 FREE TEXT

                    INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                    LAST EDITED:      MAR 21, 2000
                    HELP-PROMPT:      Answer must be 1-30 characters in
length.

90069.2121,.06    PARAM06                  0;6 FREE TEXT

                    INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                    LAST EDITED:      MAR 21, 2000
                    HELP-PROMPT:      Answer must be 1-30 characters in
length.

90069.2121,.07    PARAM07                  0;7 FREE TEXT

                    INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                    LAST EDITED:      MAR 21, 2000
                    HELP-PROMPT:      Answer must be 1-30 characters in
length.

90069.2121,.08    PARAM08                  0;8 FREE TEXT

                    INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                    LAST EDITED:      MAR 21, 2000
                    HELP-PROMPT:      Answer must be 1-30 characters in
length.
```

```
90069,51      PACKAGE                51;0 POINTER Multiple #90069.051

              DESCRIPTION:      Allows user to enter a package to establish
the
                                user's defaults for the GUI PATIENT CHART
                                application.

              TECHNICAL DESCR:
                                Pointer to the RPMS PACKAGE file, dinumed.


90069.051,.01   PACKAGE                0;1 POINTER TO PACKAGE FILE (#9.4)
                                   (Multiply asked)

                INPUT TRANSFORM:  S DINUM=X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Enter PACKAGE name
                DESCRIPTION:      Allows user to enter packages to set package
                                  defaults for an USER when using the GUI
                                  Patient Chart application.

                NOTES:            XXXX--CAN'T BE ALTERED EXCEPT BY PROGRAMMER

                CROSS-REFERENCE:  90069.051^B
                                  1)= S
^BGUUSER(DA(1),51,"B",$E(X,1,30),DA)=""
                                  2)= K ^BGUUSER(DA(1),51,"B",$E(X,1,30),DA)


90069.051,.02   PARAM02                0;2 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 1-30 characters in length.

90069.051,.03   PARAM03                0;3 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 1-30 characters in length.

90069.051,.04   PARAM04                0;4 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 1-30 characters in length.

90069.051,.05   PARAM05                0;5 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 1-30 characters in length.

90069.051,.06   PARAM06                0;6 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 1-30 characters in length.

90069.051,.07   PARAM07                0;7 FREE TEXT
```

```
                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 1-30 characters in length.

90069.051,.08   PARAM08                0;8 FREE TEXT

                INPUT TRANSFORM:  K:$L(X)>30!($L(X)<1) X
                LAST EDITED:      FEB 17, 2000
                HELP-PROMPT:      Answer must be 1-30 characters in length.



      FILES POINTED TO                      FIELDS

INSTITUTION (#4)                  INSTITUTION:INSTITUTION (#.01)

NEW PERSON (#200)                 NAME (#.01)

PACKAGE (#9.4)                    PACKAGE:PACKAGE (#.01)


INPUT TEMPLATE(S):

PRINT TEMPLATE(S):

SORT TEMPLATE(S):

FORM(S)/BLOCK(S):
```

21.0 Appendix C: Infrastructure (BGULib.dll, Clib.dll, BGUMDao.dll)

Preface

This document is designed to provide a developer with the information needed to understand and use the Indian Health Service (IHS) Client/Server Software Development Kit (SDK). Included are:

1. A detailed description of the functions available from the BGULIB DLL.
2. A detailed description of the properties and methods available from the CLIB OLE automation server.
3. A detailed description of the MDAO automation server.
4. A detailed description of the BGUMDAO DLL.
5. Instructions for installing and setting up Server software.

BGULib.DLL

The BGULib DLL contains useful functions that are available to any Visual Basic application. Before any of these functions can be used, they must be declared in a Visual Basic module, as shown in the following example.

```
Declare Function GetFileVers Lib "BGULib" ( ByVal sFileName As String) As String
```

Function GetRegistryValue(hBaseKey as long, sKeyNames as string, sValueName as string, sValue as string) as long

This function gets a registry value from the Windows 95 or Windows NT registry. The function returns the long value **ERROR_SUCCESS** if the call is successful. Otherwise, an error has occurred. If successful, the registry value is in **sValue,** which is a character 30 delimited string. The first piece of this string is the number of character 30 delimited lines returned.

Example:

```
sDelim = Chr$(30)
lBaseKey = HKEY_CURRENT_USER
sKeyNames = "Software\Indian Health Service\BGUSYNC\MRUServerData"
sValueName = "MRUServerIndex"
lRet = GetRegistryValue(lBaseKey, sKeyNames, sValueName, sValue)
If lRet = ERROR_SUCCESS Then
 sValue = Lib.GetPiece(sValue, sDelim, 2, 2)
Else
 sValue = "-1"
End If
```

Function GetWindowsType() as string

This function returns a string that identifies the current Windows version; i.e., Windows 95, Windows NT, or Windows 3.1.

Example: lblWindVersion.Caption = **GetWindowsType()**

Function GetFileVers (sFileName as string) as string

This function returns the version number of **sFileName** as a string.

Example:

```
lblDLLVersion.Caption = GetFileVers("BGULIB.dll")
lblOCX1.Caption = GetFileVers("BGUSync.ocx")
```

```
lblBGUOcx.Caption = GetFileVers("BGUOCX.ocx")
```

Function SetRegistryValueStr(hBaseKey as Long, sKeyNames as String, sValueName as String, sValue as String) as Long

This function sets the Windows 95 or Windows NT registry to the string value specified in **sValue.** The function returns the long value **ERROR_SUCCESS** if the call is successful. Otherwise, an error has occurred.

Example:

```
lBaseKey = HKEY_CURRENT_USER
sKeyNames = “Software\Indian Health Service\BPCPC\Defaults”
sValueName = “Diagnosis”
lRet = SetRegistryValueStr(lBaseKey,sKeyNames,sValueName,sValue)
if lRet <> ERROR_SUCCESS Then Lib.ErrMsg “Diagnosis Default Could Not Be Set!”
```

Clib.DLL

Clib.DLL is an **in process OLE automation server** that can be used by any Visual Basic program that references the **Library Class.** The Library Class is represented by the Clib object. A Clib object should be declared as follows in the declaration section of each Visual Basic form that uses it:

Dim Lib as New Clib

Lib is then an instance of the Clib object. The examples in this section assume that the above declaration has been made and will use the name Lib as the reference to an instance of the Clib class.

To use the Library Class, Clib.DLL needs to be on your machine and registered properly in the registry. The BGUDev installation process will do this for you. A project cannot declare and use the class as demonstrated above until the Library Class is referenced as follows:

1. Click the References option within the Tools menu.
2. Find the Library Class item in the Available References list box and select the check box next to this reference by clicking on it. This reference must be selected for the project to properly reference the class. If this item is not there, the Clib.DLL has not been registered properly.
3. Click the OK button to complete the referencing process.

Properties

AvailPhys as Long (Read Only)

This property returns the available (not used by Windows) physical memory (RAM) on the current machine in bytes.

Example: lblAvailPhysMemory.Caption = Str$(**Lib.AvailPhys**)

TotalPhys as Long (Read Only)

This property returns the total physical memory (RAM) on the current machine in bytes.

Example: lblTotalPhysMemory.Caption = Str$(**Lib.TotalPhys**)

AvailVirtual as Long (Read Only)

This property returns the available (not used by Windows) virtual memory (RAM and Disk) on the current machine in bytes.

Example: lblAvailVirtualMemory.Caption = Str$(**Lib.AvailVirtual**)

TotalVirtual as Long (Read Only)

This property returns the total virtual memory (RAM and Disk) on the current machine in bytes.

Example: lblTotalVirtualMemory.Caption = Str$(**Lib.TotalVirtual**)

MemoryLoad as Long (Read Only)

This property is the percent of memory in use on the current machine.

Example: lblMemoryLoad.Caption = Str$(**Lib.MemoryLoad**)

WindowsType as String (Read Only)

This property is the name of the current Windows operating system; i.e., Windows 95, Windows NT, or Windows 3.1.

Example: lblWindowsType.Caption = **Lib.WindowsType**

WindowsVersion as String (Read Only)

This property is the version number of the current Windows operating system as a string in the "##.##" format.

Example: lblWindowsVers.Caption = **Lib.WindowsVersion**

Methods

Function ClientHeight(hWnd as Integer) as Integer

This method returns the height of the client area (the area not including the title inside the borders) in twips of a Window or form with a handle of hWnd .

Example: nClientHeight=**Lib.ClientHeight(frmMain.hWnd)**

Function ClientWidth(hWnd as Integer) as Integer

This method returns the width of the client area (the area not including the title inside the borders) in twips of a Window or form with a handle of hWnd.

Example: nClientWidth=**Lib.ClientWidth(frmMain.hWnd)**

Function GetPiece(sStr as string,sDelim as String,iStart as Integer,iEnd as Integer) as String

This method acts like the $Piece function in MUMPS. The start and end pieces must be included in the call even if only one piece is to be retrieved.

Example: sString = **Lib.GetPiece("AB^CD","^",2,2)**
sString would equal "CD"

Sub ClearListView(hWnd as Integer)

This method clears selected items in the list box sent to it.

Example: If ckbClear.Checked=True Then **Lib.ClearListView(lvList.hWnd)**

Function ConvertDT(sDateTime as String,sFormatString as String) as String

This method converts a VA-formatted date and time (sDateTime) to an externally formatted date and time based on a format string (sFormatString) and sends back the external date and time.

Example: sString = **Lib.ConvertDT("2961224.1025","mm/dd/yy@hh:mm")**
sString would equal "12/24/96@10:25"

Sub ErrMsg(sMsg as String)

This method beeps and displays sMsg in a message box.

Example: If lRet = -1 Then **Lib.ErrMsg "No Items Have Been Chosen!"**

Sub InfoMsg(sMsg as String)

This method displays sMsg in a message box.

Example: Lib.InfoMsg "Process Complete"

Function GetLVSelItem (hWnd as Integer) as Long

This method retrieves the first item selected in a listview box. A -1 is returned if no item is selected.

Example: lRet = **Lib.GetLVSelItem(lvList.hWnd)**
If lRet = -1 then Lib.ErrMsg "No Item Selected!"

Function IncStr(sVal as String,lValue as Long,lMax as Long,lMin as Long) as String

This method increments the value of sVal by lValue within the lMax and lMin limits and returns this value as a string. This method should be used with edit controls that process numeric input only–preferably those to which a spin box is attached.

Example:

```
Private Sub SpinButton1_SpinUp()
 txtValue.Text = Lib.IncStr(txtValue.Text,5,100,5)
End Sub
```

Function DecStr(sVal as String,lValue as Long,lMax as Long,lMin as Long) as String

This method decrements the value of sVal by lValue within the lMax and lMin limits and returns this value as a string. This method should be used with edit controls that process numeric input only–preferably those to which a spin box is attached.

Example:

```
Private Sub SpinButton1_SpinDown()
 txtValue.Text = Lib.DecStr(txtValue.Text,5,100,5)
End Sub
```

Sub FormCenter(hWnd as Integer)

This routine centers a form on the screen.

Example:

```
Sub frmGetData_Load()
  Lib.FormCenter Me.hWnd
End Sub
```

Function GetYesOrNo(sPrompt as String) as Integer

This function displays a message box containing sPrompt. The message box contains a Yes and No button and returns the value of the button selected (vbYes=6,vbNo=7).

Example:

```
If Lib.GetYesOrNo("Delete Item?") = vbYes then
 DeleteItem
 Lib.InfoMsg "Item Deleted!"
Else
 Lib.InfoMsg "Item Not Deleted!"
End If
```

Function LVItemSelected(hWnd as Integer) as Boolean

This function checks a listview box to see if any items are selected and returns true or false.

Example: If **Lib.LVItemSelected(lvList.hWnd) = False** Then Lib.ErrMsg "No Item Selected!"

Function DateOk(sDate as String) as Boolean

This function checks sDate to see if it is a valid FileMan date and returns true or false. This function supports "NOW" or "TODAY" conventions, so a string like "T+3" will return true.

Example: If **Lib.DateOK("03Jul96") = False** Then Lib.ErrMsg "Improper Date Entered!"

Function DT(sDateTime as String,sFormatString as String) as String

This method converts a valid FileMan date and/or time (sDateTime) to an externally formatted date and time based on a format string (sFormatString) and returns the external date and time. This function supports "TODAY," "NOW," "MID," and "NOON" conventions. If sDateTime is not a valid FileMan date/time then the value of sDateTime is returned.

Example: sString = **Lib. DT("Today+3@Noon","mm/dd/yy@hh:nn")**
if today's date is December 24,1996 then sString would equal "12/27/96@12:00"

Function DecPlace(sNum as String) as Integer

This function returns the number of decimal places in sNum.

Example: If **Lib.DecPlace(txtValue.Text)<>2** Then Lib.ErrMsg "Two Decimal Places Required!"

Function FileExists(sFileName as String) as Boolean

This function checks to see if sFileName exists and returns true or false. The filename parameter can contain complete path information as well.

Example: If **Lib.FileExists("C:\WINDOWS\MYFILE.DAT") = True** Then Lib.InfoMsg "File Exists!"

Function InCombo(hWnd as Integer,strText as String) as Integer

This function checks to see if strText is in a combo box with a window handle of hWnd. A -1 is returned if strText is not in the combo box. The index of strText is returned if strText is in the combo box.

Example: If **Lib.InCombo(cbEntry.hWnd,"New Entry")=-1** Then cbEntry.AddItem "New Entry"

Function InList(hWnd as Integer,strText as String) as Integer

This function checks to see if strText is in a list box. The window handle of the list box is passed in the hWnd parameter. A -1 is returned if strText is not in the list box. The index of strText is returned if strText is in the list box.

Example: If **Lib.InList(lstEntry.hWnd,"New Entry") = -1** Then lstEntry.AddItem "New Entry"

Function IsInteger(sVal as String) as Boolean

This function checks sVal to see if it is an integer and returns true or false.

Example: If **Lib.IsInteger(sValue) = False** Then Lib.ErrMsg "Value Is Not An Integer!"

Function MakeUpper(sText as String,KeyAscii as Integer,bTextRequired as Integer) as Integer

This function converts text typed in an edit box to uppercase. It also processes RETURN key (ascii 13) presses. KeyAscii represents a key that was just pressed. bTextRequired tells the function if text is required in the edit box before a RETURN key can be processed. A 0 is returned if the RETURN key was pressed and successfully processed. A 13 is returned if a RETURN key was pressed but not successfully processed. If the return key was not pressed an integer representing a key is returned. The sText parameter should contain the text from the edit box for which you are calling this function, as indicated in the example below.

Example:

```
Sub txtDiagnosis_KeyPress(KeyAscii As Integer)
  KeyAscii = Lib.MakeUpper(txtDiagnosis.Text, KeyAscii, True)
  If KeyAscii = 0 Then cmdDisplay.Value = 1
End Sub
```

Function FloatOnly(sTxt as String,KeyAscii as Integer,nDecPl as Integer, nTextReq as Integer) as Integer

This function filters out all keystrokes in an edit box control except numbers, the decimal ("."), and the return key (ascii 13). KeyAscii represents the key that was just pressed. nDecimalPlace indicates the maximum decimal places allowed. If an attempt is made to enter a number beyond nDecimalPlace, a beep is sounded and the number discarded. BTextRequired tells the function if text is required in the edit box before the return key can be processed. A 0 is returned if the RETURN key was pressed and successfully processed. A 13 is returned if the RETURN key was pressed but not successfully processed or KeyAscii was not a numeric or decimal key. Otherwise, the ASCII value of the key pressed is returned. The sText parameter should contain the text from the edit box for which you are calling this function, as indicated in the example below.

Example:

```
Sub txtAmt_KeyPress(KeyAscii As Integer)
  KeyAscii = Lib.FloatOnly(txtDays.Text, KeyAscii, 2,True)
  If KeyAscii = 0 Then cmdDisplay.Value = 1
End Sub
```

Function NumberOnly(sText as String,KeyAscii as Integer,bTextRequired asInteger) as Integer

This function filters out all keystrokes in an edit box control except numbers and the RETURN key (ASCII 13). KeyAscii represents the key that was just pressed. bTextRequired tells the function if text is required in the edit box before the RETURN key can be processed. A 0 is returned if the RETURN key was pressed and successfully processed. A 13 is returned if the RETURN key was pressed but not successfully processed or KeyAscii was not a numeric key. Otherwise, the ASCII value of the key pressed is returned. The sText parameter should contain the text from the edit box for which you are calling this function, as indicated in the example below.

Example:

```
Sub txtDays_KeyPress(KeyAscii As Integer)
  KeyAscii = Lib.NumberOnly(txtDays.Text, KeyAscii, True)
  If KeyAscii = 0 Then cmdDisplay.Value = 1
End Sub
```

Function PieceNo(sStr as String,sDelim as String) as Integer

This function returns the number of pieces in sStr delimited by sDelim.

Example: I = **Lib.PieceNo(sData,”^”)**

```
For n=1 to i
 Lib.sPiece = Lib.GetPiece(sData,”^”,n,n)
Next n
```

Function ReplaceStr(sSource asString,sReplace asString,sReplaceWith as String) as String

This function returns a string in which all occurrences of the string sReplace have been replaced with the string sReplaceWith within the string sSource.

Example: sSource = “abcabdabe”

```
      sReplace = “ab”
  sReplaceWith = “yz”
  sStr=Lib.ReplaceStr(sSource,sReplace,sReplaceWith)
```

sStr in the above example would be “yzcyzdyze”

Sub SelectText(hWnd As Integer)

This routine selects text in an edit box with a window handle of hWnd.

Example: In this example, when the txtEntry edit box gets the focus, its text is selected.

```
Sub txtEntry_GotFocus()
 Lib.SelectText txtEntry.hWnd
End Sub
```

Sub SetListHScroll(hWnd asInteger, control,iWidth as Integer)

This routine sets up a horizontal scroll bar on the list box with a window handle of hWnd. In Windows 95 or Windows NT, if you need a horizontally scrolling list box with more capabilities than a list box, use a listview box instead.

Sub SetListTabs(hWnd as Integer,cTabs() as Long)

This routine sets up tab stops in the list box whose window handle is hWnd. In Windows 95 or Windows NT use the listview box instead.

21.1.1.1

Function TR(x asString,CHFRM as String,CHTO as String) as String

This function corresponds to the $TR function in MUMPS. It returns a string that has changed all characters of CHFRM in X to corresponding characters in CHTO.

Example: The following example converts lower-case characters in sSource to upper case.

sStr=**Lib.TR(sSource,”abcdefghijklmnopqrstuvwxyz”,”ABCDEFGHIJKLMNOPQRSTUVWXYZ”)**

Function TimeOK(sTime as String) as Boolean

This function checks sTime to see if it is a valid time and returns true or false. This function supports FileMan time conventions so “MID” and “NOON” will be returned as true with this function.

Example: If **Lib.TimeOK(sTime) =False** Then Lib.ErrMsg “Improper Time Entered!”

Function GetFileVersion(sFileName as String) as String

This function returns the version of the file sent to it. The parameter sFileName can contain the drive path and filename.

Example: lblVersion.Caption = **Lib.GetFileVersion("C:\WINDOWS\SYSTEM\BGULIB.DLL")**

Function GetFreeDiskSpace(sRootName as String) as Single

This function gets the free space left on the drive specified by sRootName in megabytes. The sRootname parameter is the root directory of the drive for which you wish to get the free space for; e.g., C:\

Example: fSpace = **Lib.GetFreeDiskSpace("C:\")**

Function GetRegVal(hBaseKey as Long,sKeyNames as String,sValueName as String) as String

This function returns a string value from the registry.

Example:

```
lBaseKey = HKEY_CURRENT_USER
sKeyNames = "Software\Indian Health Service\BPCPC\Defaults"
sValueName = "Diagnosis"
sValue = Lib.GetRegVal(lBaseKey,sKeyNames,sValueName)
```

Function SetRegValStr(hBaseKy as Long,sKyNames as String,sValName as String, sVal as String) as Long

This function sets a string value in the registry. It returns a long value. If this value is not ERR_SUCCESS, then an error occurred and the value was not written.

Example:

```
lBaseKey = HKEY_CURRENT_USER
sKeyNames = "Software\Indian Health Service\BPCPC\Defaults"
sValueName = "Diagnosis"
sValue = "1"
lRet = Lib.SetRegValStr(lBaseKey, sKeyNames, sValueName, sValue)
if lRet <> ERROR_SUCCESS then Lib.ErrMsg "Error Occurred! Value Not Written."
```

Function LineCount(hWind as Integer) as Long

This function returns the number of lines in a multiline edit box.

Example: nCount = **Lib.LineCount(txtEdit box.hWnd)**

Function GetLine(hWind as Integer,lLineNo as Long) as String

This function returns a specific line in a multiline edit box.

Example:

```
nCount = Lib.LineCount(txtEdit box.hWnd)
for I=1 to nCount
 sLine(I) = Lib.GetLine(txtEdit box.hWnd,I)
next I
```

BGUMDAO.DLL

The **BGUMDAO.dll** file is an **in process Active-X Automation Server** used by a Visual Basic program that references it. This automation server exposes two objects: the **Server** object and the **StrMap** object.

Referencing BGUMDAO.DLL in VB4

1. Click on the **References** option within the **Tools** menu.
2. Find the **BGUMDAO 1.0 Library** item in the **Available References** listbox and check the checkbox next to this reference by clicking on it. This reference must be checked for the project to properly reference BGUMDAO.DLL. If this item is not there, more than likely the BGUMDAO.dll has not be registered properly.
3. Click on the **OK** button to complete the referencing process.

Referencing BGUMDAO.DLL in VB5 or VB6

1. Click the **References** option within the **Project** menu.
2. Find the **BGUMDAO 1.0 Library** item in the **Available References** listbox and check the checkbox next to this reference by clicking on it. This reference must be checked for the project to properly reference BGUMDAO.DLL. If this item is not there, more than likely the BGUMDAO.dll has not be registered properly.
3. Click on the **OK** button to complete the referencing process.

The Server Object

The Server object contains methods and properties which pertain to managing a connection to a MUMPS database server. It's methods and properties are encapsulated by the CServer object contained in MDAO.DLL ,so if MDAO.DLL is referenced, there is no need to use the Server object.

Declaring The Server Object

The Server object can be declared locally using the **Dim** keyword or globally using the **Public** key word. It can be declared in one of two ways.

Dim objServer as New Server Or Dim objServer as Server

When the **New** keyword is used as in the first example, an instance of the object is created immediately. If the New keyword is not used in the declaration, an instance of the object must be created before it is used as follows**: Set objServer = New Server**

Server Properties

AccessCode As String (Write Only)

This property is the access code value to connect to the server. If it or the VerifyCode property is a null string, an access/verify dialog box is popped up by the object

Example: objServer.AccessCode = txtAccessCode.text

VerifyCode As String (Write Only)

This property is the verify code value to connect to the server. If it or the AccessCode property is a null string, an access/verify dialog box is popped up by the object.

Example: **objServer.VerifyCode** = txtVerifyCode.text

IsOpen As Boolean (Read Only)

This property is used to determine if the connection with the server is open or not.

Example: If **objServer.IsOpen** then DoSub

ServerPort As Long

This property holds the port number on which a connection is requested. It normally defaults to 8000 but can be set to another value by the program before a connection request is made.

Example: lblServerPort.Caption = Str$(**objServer.ServerPort**)

Server As String

This property is the Name or the IP address of the server to which the connection will be made. If it is a null string, a server address dialog box is popped up by the object.

Example: **objServer.Server** = txtAddress.text

UserName As String (Read Only)

This property contains the user name of the person logged onto the server. This property is set by the object when connection to the server is successful.

Example: lblUser.caption = **objServer.UserName**

DUZ As String (Read Only)

This property contains the IEN (Internal Entry Number) of the person logged onto the server. This property is set by the object when connection to the server is successful.

Example: lblDUZ.caption = **objServer.DUZ**

DUZ2 As String (Read Only)

This property contains the IEN (Internal Entry Number) of the site onto which the user is logged.

Example: lblDUZ2.caption = **objServer.DUZ2**

Facility As String (Read Only)

This property contains the facility name for the server onto which you are logged. It is set by the object after a successful connection is made to the server.

Example: lblLogInFacility.Caption = **objServer.Facility**

LocalAddress As String (Read Only)

This property contains the IP address of the current client machine connected to the server. It is set by the object after a successful connection is made.

Example : lblLocalAddress.Caption = **objServer.LocalAddress**

CardAddress (Read Only)

This property contains the address code for the ethernet card installed in the client's machine. The object gets this address from the client's registry when connection to the server is successful.
Example: `lblEthernetAddress.caption = **objServer.CardAddress**`

ConnectedPort As Long (Read Only)

This property contains the port number which the server assigned to the object for communication with it . It is set after connection to server is successful.
Example: lblConnectedPort.Caption = **objServer.ConnectedPort**

IHSAppID As String

This property contains the Application ID for the client program in use.
Example: lblApplicationID.caption = **objServer.IHSAppID**

UCI As String

This property is used strictly for program testing and debugging purposes. Normally a handler is jobbed in the UCI where the listener resides and this would be the case at a real site. However, there may be times in development when necessary data on the server resides on other UCIs than the listener's and instead of shutting down the listener in one UCI and starting it up again in another or installing extra data and routines in all the UCIs, this property can be set with the UCI and Volume Group for which you want a handler to be jobbed.
Example: objServer.UCI = "LBP,DEV"

AutoConnect As Boolean

This property enables or disables autoconnection to the server. If it is set to true and another program on the client is connected to the server, the current program will connect automatically to the server without prompting for access/verify code. If it is set to false, the user will have to enter access/verify codes every time an application is started.
Example: **objServer.AutoConnect** = True

Connections As integer (Read Only)

This property keeps track of the number of current connections to the server . Only those programs that have set the AutoConnect property to true will be counted.
Example: lblConnections.Caption = Str$(**objServer.Connections**)

DeveloperMode as Boolean

This property should only be used in development. When it is set to true, the connection count kept in the registry never goes below one. Therefore, all applications that have AutoConnect set to true will automatically connect to the server without having to enter an access/verify code.
Example: objServer.DeveloperMode = True

Error as String (Read Only)

This property returns the text of the last error. Reading this property automatically clears it. So using this property as follows will not work:
If objServer.Error<>"" then Lib.ErrMsg objServer.Error ← **Nothing Will Be Displayed**
Instead set a string variable to the property and then display the variable as shown below:

```
    sErr=objServer. Error ←Put the error in a string variable
    If sErr<>"" then Lib.ErrMsg sErr ← Display the variable if
necessary
```

Server Methods

Function Connect() as Boolean

This method is used to request a connection to the server. If it returns true the request was successful; otherwise, the connection failed.

Example:

```
bRet = objServer.Connect
If bRet = True Then
sUser = objServer.UserName
sAddress = objServer.Server
panStatus.Caption = sUser & " In " & objServer.Facility
cbServer.SetFocus
Else
Lib.ErrMsg "Connection Failed!"
End If
```

Function Disconnect() as Boolean

This method is used to close the connection to a server. It returns true if the close process was successful; otherwise, the close process failed.
Example: If objServer.IsOpen Then bRet = **objServer.Disconnect**

Function Execute(sString as String) as String

The Execute method sends an RPC string synchronously to the server. An RPC string consists of an RPC name field followed by the parameters to the call delimited by "^"; for example, **"BGU GETDIAGNOSISLIST^DIAB^50^^".** The return value is a character 30 delimited string. The first piece is the number of records following in the string. If this piece is empty or "-1," then an error has occurred and is in the second piece.

Example:

```
sSendString = "BPC GETDIAGNOSISLIST^DIAB^50^^"
Screen.MousePointer = CR_HOURGLASS
sResult = objServer.Execute(sSendString)
Screen.MousePointer = CR_DEFAULT

If sResult = "" Then
```

```
      txtResult.Text = “No Data Returned!”
    Exit Sub
    End If
    If Lib.GetPiece(sResult,Chr$(30),1,1)=”-1” Then
      TxtResult.Text =  Lib.GetPiece(sResult,Chr$(30),2,2)
    End If
```

Sub ClearConnections()

This method clears the connection data kept in the registry and sets the connection count to zero. It can only be called when the object is disconnected.

Example:

```
Private Sub cmdClear_Click()
objServer.Disconnect
objServer.ClearConnections
cmdSet_Click
End Sub
```

Function EncriptString(sString As String) As String

This method is used to encript a string entered as an access/verify code

Example:

```
Private Function CkVerify() As Boolean
  Dim bRet As Boolean
  Dim sVerify As String

  bRet = False
  sVerify = objS.EncriptString(txtVerify.Text)
  If sVerify = sVerifyCode Then bRet = True
  CkVerify = bRet
End Function
```

The StrMap Object

The StrMap object allows the creation of the equivalent of a string subscripted array. Though similar to a MUMPS array, it is neither as flexible or as powerful. In addition, because it is not a btree structured object, data is not sorted by subscript as MUMPS arrays are. The data contained in each StrMap node is a string.

Declaring The StrMap Object

The StrMap object can be declared locally using the **Dim** keyword or globally using the **Public** key word. It can be declared in one of two ways.

Dim mapData as New StrMap Or Dim mapData as StrMap

When the **New** keyword is used as in the first example, an instance of the object is created immediately. If the New keyword is not used in the declaration, an instance of the object must be created before it is used as follows**: Set mapData = New StrMap**

StrMap Properties

Overlay() As Boolean

THE OVERLAY PROPERTY TELLS THE OBJECT WHETHER TO OVERLAY OR ADD DATA TO EXISTING DATA WHEN SETTING A STRMAP VALUE. OVERLAY DEFAULTS TO TRUE. IF OVERLAY IS FALSE, NEW DATA IS ADDED AT THE END OF CURRENT DATA DELIMITED BY A CARRIAGE RETURN LINEFEED ($C(13,10)).

Example:

```
If sFieldType = "M" Or sFieldType = "W" Then
    mapData.Overlay = False
    mapData(sKey) = sNewValue
    mapData.Overlay = True
Else
     mapData(sFldName) = sNewValue
End If
```

Count() As Long (Read Only)

The Count property returns the number of items in the StrMap object being referenced.

Example:

```
N = mapData.Count
If N > 0 Then
  Redim sValue(1 To N)
    Redim sKey(1 To N)
  For I = 1 To N
    sValue(I) = mapData.ValByNo(I)
    sKey(I) = mapData.KeyByNo(I)
  Next
End If
```

ValByNo(nIndex As Long) As String (Read Only)

The ValByNo property returns the value of a StrMap item referenced by it's number. **See the Count property example.**

KeyByNo(nIndex As Long) As String (Read Only)

The KeyByNo property returns the key (subscript) of a StrMap item referenced by it's number. **See the Count property example.**

Value(sKey As String) As String

The Value property sets or returns the value of a StrMap item referenced by it's key (subscript). If the key value does not exist when getting a value, a null string ("") is returned. This property is the default property for the StrMap object. Therefore it can be used in one of two ways: **mapData.Value("Key")="Value" or mapData("Key")="Value"**

Example:

sVal = mapData(sKey) ← **Alternate: sVal = mapData.Value(sKey)**
If sVal = "" Then **mapData(sKey) = sNew** ← **Alternate : mapData.Value(sKey) = sNew**

KeySetVal(nIndex As Long) As String (Read Only)

The KeySetVal property returns a value from a KeySet referenced by it's number. **See the CreateKeySet method.**

KeySetKey(nIndex As Long) As String (Read Only)

The KeySetKey property returns the key from a KeySet referenced by it's number. **See the CreateKeySet method.**

StrMap Methods

Sub Clear()

The Clear method removes all the items from a StrMap object.

Example: If bNewData Then **mapData.Clear**

Sub Remove(sKey As String)

The Remove method removes an item from a StrMap based on it's key. If the key doesn't exist, the method is ignored.

Example: If bRemove Then **mapData.Remove(sKey)**

Function CreateKeySet(sKey As String) As Long

The CreateKeySet method creates a KeySet based on a string value (sKey) and returns the number of items in the KeySet. A KeySet is a subset of StrMap items whose keys start with a common string. If no keys in the StrMap object start with the string, then the method returns a zero. As the example shows, the CreateFieldSet method of the CMFile object in the MDAO.DLL uses the CreateKeySet method to create multiple and word processing keysets.

Example:

```
Public Function CreateFieldSet(sFieldName As String) As Integer
  Dim nRet As Integer
  Dim sKey As String

  nRet = 0
  m_bIENsSet = False
  m_sIENsType = ""
  If m_nValIndex > 0 Then
    m_sIENsType = Lib.GetPiece(m_mapFldDef(UCase$(sFieldName)), U, 5, 5)
    sKey = UCase$(sFieldName) & "," & m_nValIndex & ","
    If m_sIENsType = "M" Then
      nRet = m_mapMultVal.CreateKeySet(sKey)
    ElseIf m_sIENsType = "W" Then
      nRet = m_mapWPVal.CreateKeySet(sKey)
    End If
  End If
  If nRet > 0 Then m_bIENsSet = True
  CreateFieldSet = nRet
End Function
```

22.0 Appendix D: IHS Telnet Control Version 1.2 (BGUTN.OCX)

Introduction

BGUTN.OCX is a 32-bit **Active-X Control** that can be used in any programming environment that creates 32-bit Windows programs and supports 32-bit Active-X Controls. It provides properties, methods, and events pertaining to establishing and maintaining a Telnet session thru TCP/IP to a chosen Telnet server. Before BGUTN can be used BGUTN.OCX must be present and registered properly on your machine. To use the BGUTN control in your VB 5 project, it must be added to the toolbox as follows:

1. Click the **Components** option within the **Project** menu.
2. Find the **IHS Telnet Active X Control** item in the **Controls** listbox and check the checkbox next to it. If this item is not there, more than likely the BGUTN.OCX has not be registered properly.
3. Click on the OK button to complete the process.

Once the BGUTN control is added to the toolbox, it can be dragged onto a VB form like any other component in the toolbox. The examples in this section assume that a BGUTN object named **TNet** has been added to the form.

Properties

AutoSize as Boolean

This property determines whether the viewing area of the BGUTN control will automatically size at design or run time based on the Columns and Rows and Font properties. It defaults to True. The AutoSize property can only be set before the BGUTN control is connected. Otherwise, the setting is ignored.

Example: `TNet.AutoSize = True`

BackColor as OLE_COLOR

This property determines the background color of the control. It defaults to white (RGB(255,255,255)). BackColor can only be set before the control is connected or it is ignored.

Example: `TNet.BackColor = RGB(0,0,255)` ← Set color to blue

BoldColor as OLE_COLOR

This property determines the color of bolded text at run time. It defaults to black (RGB(0,0,0)) . BoldColor can only be set before the BGUTN control is connected. Otherwise, the setting is ignored.

Example: `TNet.BoldColor = RGB(255,255,255)` ← Sets bold color to white

BufferText as String (Read Only)

This property returns text in the control's text buffer. Each line of text is delimited by a CRLF (Ascii 13,Ascii 10).

Example:

```
sText = TNet.BufferText
n = TNet.LineCount
For i = 1 To n
  sS = Lib.GetPiece(sText, vbCrLf, i, i)
  Printer.Print sS
Next
```

Columns as Integer

This property determines the maximum number of characters displayed on a line. If a line has more characters that this, it is wrapped to the next line. The Columns property defaults to 80. It can only be set when the control is not connected. Otherwise, the setting is ignored.

Example: `TNet.Columns = 132`

Font as Font

THIS PROPERTY DETERMINES THE FONT OF THE CHARACTERS DISPLAYED IN THE CONTROL. THE FONT NAME DEFAULTS TO COURIER NEW, THE FONT SIZE TO 9, AND THE FONT STYLE TO NORMAL. THIS PROPERTY CAN ONLY BE SET WHEN THE CONTROL IS NOT CONNECTED, OR IT IS IGNORED.

Example:

```
TNet.Font.Name = ”DECVT”
TNet.Font.Size = 9
```

ForeColor as OLE_COLOR

This property determines the foreground color of the control. It defaults to blue (RGB(0,0,255)). ForeColor can only be set before the control is connected or it is ignored.

Example: `TNet.ForeColor = RGB(0,255,0)` ← Set color to green

Host as String

This property determines the host to which the BGUTN control will connect. It can be a DNS name or TCP/IP address. Connection will fail if this property is not set correctly.

Example: `TNet.Host = "161.223.1.12”`

IsTextSelected as Boolean (Read Only)

This property determines whether or not text has been selected within the textbuffer.

Example: `If TNet.IsTextSelected Then PrintSelectedText`

LineCount as Integer (Read Only)

This property returns the number of lines currently in the control's text buffer.

Example:

```
sText = TNet.BufferText
n = TNet.LineCount
For i = 1 To n
  sS = Lib.GetPiece(sText, vbCrLf, i, i)
  Printer.Print sS
Next
```

Lines as Integer

This property determines the maximum number of lines in the control's text buffer. The Lines property defaults to 500. When the number of lines in the buffer exceed this value, the buffer is cleared and data is stored in the buffer starting at the first line. This value must be set while the control is not connected or it is ignored. The Lines property cannot be set lower than 50. If the value is below 50, it will be set to 50.

Example: `TNet.Lines = 1000`

Port as Integer

This property determines the port which to connect. The Port property defaults to 23. This value must be set while the control is not connected or it is ignored.

Example: `TNet.Port = 25`

Rows as Integer

This property determines the number of lines that can appear at one time in the display area of the control. The text in the display area scrolls down when the number of lines to be displayed exceed the Rows property.

Example: `TNet.Rows = 25`

RVBoldColor as OLE_COLOR

This property determines the color of text in the control's display area that is reverse bolded. It defaults to white (RGB(255,255,255)). The RVBoldColor property can only be set when the control is disconnected or the setting is ignored.

Example: `TNet.RVBoldColor  = RGB(0,0,0)` ←Sets color to black

ScreenText as String (Read Only)

This property returns text from the current screen. Each line of text is delimited by a CRLF (Ascii 13,Ascii 10).

Example:

```
sText = TNet.ScreenText
n = Lib.PieceNo(sText,vbCrLf)
For i = 1 To n
  sS = Lib.GetPiece(sText, vbCrLf, i, i)
  Printer.Print sS
Next
```

SelectedBackColor as OLE_COLOR

This property determines the background color of selected text in the control's display. It defaults to black (RGB(0,0,0)). The SelectedBackColor property can only be set when the control is disconnected or the setting is ignored.

Example: `TNet.SelectedBackColor  = RGB(0,0,255)` ←Set color to blue

SelectedForeColor as OLE_COLOR

This property determines the text color of selected text in the control's display. It defaults to white (RGB(255,255,255)). The SelectedForeColor property can only be set when the control is disconnected or the setting is ignored.

Example: `TNet.SelectedForeColor  = RGB(255,0,0)` ←Set color to red

SelectedText as String (Read Only)

This property returns selected text from the control's text buffer. Each line of text is delimited by a CRLF (Ascii 13,Ascii 10).

Example:

```
sText = TNet.SelectedText
n = Lib.PieceNo(sText,vbCrLf)
For i = 1 To n
  sS = Lib.GetPiece(sText, vbCrLf, i, i)
  Printer.Print sS
Next
```

Methods

Function Connect() as Boolean

This method is used to request a connection to the Telnet server. If it returns true the request was successful; otherwise, the connection failed.

Example:

```
If Not TNet.Connect Then
       Lib.ErrMsg "Connection Failed!"
       End
End If
```

Function Disconnect() as Boolean

This method is used to disconnect from the Telnet server. Currently this method always returns true.

Example: `bRet = TNet.Disconnect`

Sub ClearSelection()

This method clears selected text in the control's text buffer. If no text is selected, it is ignored.

Example: `TNet.ClearSelection`

Events

Sub OnSelect(ByVal nLines as Integer)

This event is fired after text selection is complete. The parameter nLines contains the number of lines selected.

Example:

```
Private Sub TNet_OnSelect(ByVal nLines As Integer)
      If nLines > 0 Then mnuPrintSelection.Enabled = True
End Sub
```

Sub OnUnSelect()

This event is fired after all selected text in the control has been cleared.

Example:

```
Private Sub TNet_OnUnSelect()
      MnuPrintSelection.Enabled = False
End Sub
```

23.0 Appendix E: GENLIST V 1.2

Introduction

Genlist is a utility that retrieves data from a FileMan defined database by sending it a list of mandatory and optional parameters. This tool provides a developer versed in FileMan a familiar tool set for retrieving items needed in creating a mumps or Windows application.
The utility is comprised of a set of routines in the BGULIST name space. This is a silent utility -- only strings of data are returned, prompts or messages are not displayed.

Implementation

There are three approaches in using Genlist: 1) calling the remote procedure call BGU GENLIST, 2) using MDAO (Mumps Data Access Object), and 3) doing the routine ^BGULIST directly. The most commonly used are MDAO and the remote procedure call. The remote procedure approach is described in this document. Refer to the ***MUMPS Data Access Object*** document for any questions regarding the implementation or the use of MDAO. The difference between doing ^BGULIST directly and calling the BGU GENLIST remote procedure call is that the programmer is responsible for setting the environment variables, e.g. U = "^", and invoking ^BGULIST passing the necessary parameters. The only requirement using the RPC is passing a list of parameters.

Parameters

The following parameters will be discussed in the order they are defined in ^BGULIST (remote procedure BGU GENLIST). Actually, the first parameter is **ARRAY**, but that is supplied by the routine that makes the remote procedure calls, ^BGUTCPH. **ARRAY** is where the data is returned. Note: if BGULIST is called directly then this parameter would be the first passed in the call.

File

The primary file providing the data, e.g. 2 for ^DPT. A file name or number can be used, e.g. VA PATIENT or 2.

Ien

The Internal Entry Number - If this is set fields specific only to this record are returned. If it is null then the program is in list mode – multiple records will be returned that satisfy the criteria set by the parameters.

More

This flag is set on (=1) if you want the data to start from where it left off on the previous call. If this flag
is not set then the program will start at the beginning determined by the value of **BEGIN.**

Cross-reference

The cross-reference that is used to access the main file (or Lookup routine i.e. SC:TAG|ROUTINE).

This is usually a letter such as “B” or a string. Whatever the value is will be used as the top subscript in the primary file. The file is then traversed and all **IEN**s with the associated records is returned.

Maximum

The number of records to return on a call (default = 25).

Begin

The starting position in the cross-reference file if one is specified, otherwise the starting point in the main file. This parameter is only used when in list mode (an IEN was not specified).

End

The opposite of **BEGIN** (BGUBEGIN and BGUEND are used inclusively).

Fields

A list of FileMan field numbers separated by commas. If a field is from the primary file just the field number is entered. If a field was returned using a lookup into another file then the field number is prefixed with the file number and a semicolon, e.g. 60;.01.

Direction

Controls the direction that the primary file is scanned. F – forward, B – backward (forward is default).

Screen

Mumps executable code used for screening data. If this option is set only data verified by this screen will be returned.

Unique Identification String

This parameter can be set to any unique identifier, e.g. $J. This parameter is supplied by the MDAO using the client machine’s network card address plus the current time including seconds and thousandths of a second.

Examples

The following examples will show strings that would be sent from a Windows client application to a Mumps-FileMan database server and a description of the returned data. These strings will show the actual values corresponding to the parameters that control Genlist. In every case the remote procedure is **BGU GENLIST**. The delimiter is the caret: ^.

This example splits up the string for display purposes only. The ellipses are used to illustrate where the splits occur:

BGU GENLIST^2^^1^B^^ADAM^ADAM~^ . . .
.001,.01,.02,.03,.09,F9000001:.001~""41""~DUZ(2)~""0"":4101!.02 . . .
^^$E(BGUV(2,.01),1,$L(BGUBEGIN))=BGUBEGIN

This example shows that a file number can be used interchangeably with a file name for the **FILE** parameter (2 or PATIENT). The 1 is the value for the **MORE** parameter, which indicates that subsequent calls made to Genlist are to return data starting from where the previous calls had left off. The B cross-reference is traversed, and **START** equals ADAM and **END** equals ADAM~. The effect this will have is Genlist will only return IENs that correspond to the entries in the cross-reference that are in the range of last names that starts with ADAM and ends with ADAM at the beginning of the last name, such as Adamson.

The fields requested are **.001,.01,.02,.03,.09,F9000001:.001~""41""~DUZ(2)~""0"":4101!.02** .

By convention Genlist uses .001 to indicate the value of the primary key (IEN) of the file, and .0001 as the value of the subscript in the cross-reference, which in this case would be a patient's name. .01,.02,etc., are actual FileMan field numbers. .01 is the Patient Name, .02 is Sex, .03 is DOB, and so on. Further down the string is **F9000001:.001~""41""~DUZ(2)~""0"":4101!.02** . This defines a lookup to another file, 9000001 (the IHS Patient file) using .001 (the IEN) the literal value 41, the value of variable DUZ(2) (the site code) and literal value 0, as subscripts into this file, and returns the FileMan field .02 which is a part of the Multiple field 4101. Note that literal values have to be enclosed in a pair of quotes. This is required in order for Mumps to interpret these values as literals.

The syntax Genlist uses to recognize a multiple is number!number…!number. The first number is the field number for the multiple-field defined in the primary file. The second field is a field defined in the sub-file (the multiple). This construct can define any field in any file regardless of how many dimensions deep the file is defined. Each piece of the field delimited by the "!" indicates a level in the file. So in the case of the field 4101!.02 the data referenced is in the second level of the file.

The string **$E(BGUV(2,.01),1,$L(BGUBEGIN))=BGUBEGIN** is the value for **SCREEN**. Genlist will build a string that will set the value of a variable equal to the Boolean value of the execution of this statement. If the value is false the data for the corresponding record will not be returned. If **SCREEN** isn't set then all encountered records are returned.

The following shows what Genlist returns as a result of the above call:

```
^TMP("LIST","32","0","0")="25^1"
^TMP("LIST","32","1","1")="¯473\0200\025ADAMS,AMANDA^F^2780822^^^^^^04738047
3"
^TMP("LIST","32","1","2")=".001~"41"~DUZ(2)~"0"->F9000001:473,41,4585,0\025^101500"
^TMP("LIST","32","1","3")="SF:.001\025473"
^TMP("LIST","32","2","1")="¯781\0200\025ADAMS,ANDY^M^2900103^^^^^^078190781"
^TMP("LIST","32","2","2")=".001~"41"~DUZ(2)~"0"->F9000001:781,41,4585,0\025^101926"
^TMP("LIST","32","2","3")="SF:.001\025781"
```

```
^TMP("LIST","32","3","1")="¯831\0200\025ADAMS,BARNEY^M^2900808^^^^^^083150831
"
^TMP("LIST","32","3","2")=".001~"41"~DUZ(2)~"0"->F9000001:831,41,4585,0\025^101988"
```

The first record is the total number of records returned in this call , with a flag indicating there are more records to come in this record set. The primary delimiter between all records is the character 175. I used ^%GL to display this global and it uses the convention of a back slash followed by three digits to represent the non-visual characters (less than 32 in the ASCII character set). So in this example \020 is control-T, and \025 is control-Y. Because of this approach sometimes it is hard to see in the displays where the control characters end and the text data begins. Our terminal emulator software represents the record delimiter, graphic character 175, as a macron (looks like a raised underscore). These delimiters were chosen since it would be difficult for them to be entered inadvertently with text data. The \025 strings seen interspersed in the data are control-Y characters, the primary delimiter used in the record segments to separate the data from its corresponding global node subscript. Control-T represented by \020 separates the IEN from its data.

All of the data that are returned for the requested fields are grouped together in separate strings, one for each global node that the requested fields belong to. The data is in the order that it occurred in within the node where it was extracted. If the fields corresponding to data in the first, third and fifth pieces of the same global node are requested then the data string that will be returned for these fields will have only the first, third and fifth pieces set. This enables the algorithm that parses the data from the string to the field values to be very straightforward and efficient. The algorithm that updates the FileMan files relies on this same principle.

BGU GENLIST^ PATIENT ALLERGIES^2^^^^^^.001,.01,1

The primary file is the Allergies file and in this case only one entry is targeted – entry number 2. (i.e. IEN = 2) In the field list .001 is the value for the IEN, .01 is a pointer to VA PATIENT and 1 is a variable pointer. The actual data stored in the patient file for the pointer fields .01 and 1 are "1" and "3;GMRD(120.82,", respectively., but what will be returned as values for the corresponding fields is "WATERMAN, RAE" and "CHOCOLATE." The reason for this is that GENLIST will return the resolved value of a pointer unless a "-P" is added to the pointer field in the field list. Since "-P" was not specified Genlist used the pointer field's definition to find which file it pointed to, and returned the value in the .01 field position stored in that file using the pointer value as the subscript. If there had been another pointer stored at this point the process would continue until a non-pointer value was returned.

Variable pointers are very similar to pointers except the file it points to is not in the data dictionary but in the data itself. So in this case Genlist looks in the file referenced in the data, "^GMRD(120.82 ," and uses the first part of the field , "3", as the subscript.

```
^TMP("LIST","32","0","0")="1"
^TMP("LIST","32","1","1")="¯2\0200\025WATERMAN,RAE^^CHOCOLATE"
^TMP("LIST","32","1","2")="SF:.001\0252"
```

BGU GENLIST^VA PATIENT^1^^^^^^.001,.01,.02,.03,.033,.09

Genlist handles a computed field the same way FileMan does. In this example field .033, age, is a computed field. Genlist executes the computed field logic that is defined in the data dictionary, and then returns the value. So in this example the patient's birthday is 11/10/70 (2701110) and 27 is the computed value returned for the age:

```
^TMP("LIST","41","0","0")="1"
^TMP("LIST","41","1","1")="¯1\0200\025WATERMAN,RAE^F^2701110^^^^^^000120001"
^TMP("LIST","41","1","2")="SF:.033\02527"

    ^TMP("LIST","41","1","3")="SF:.001\0251"
```

Above is the global that stores the data that is returned by Genlist. Notice that the line with the underlined 27 is prefixed with "SF:". The "SF" indicates that the field that follows, .033, is a special field. This convention is followed whenever a field that is not mapped to a specific node and piece in a file is returned.